

PART 2 – Columnar Database

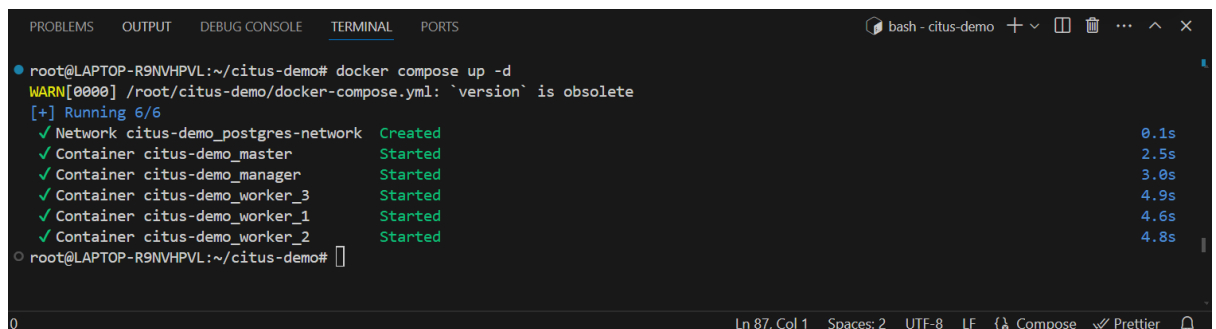
TASK

1. Jalankan Citus di komputer lokal dengan menggunakan docker compose

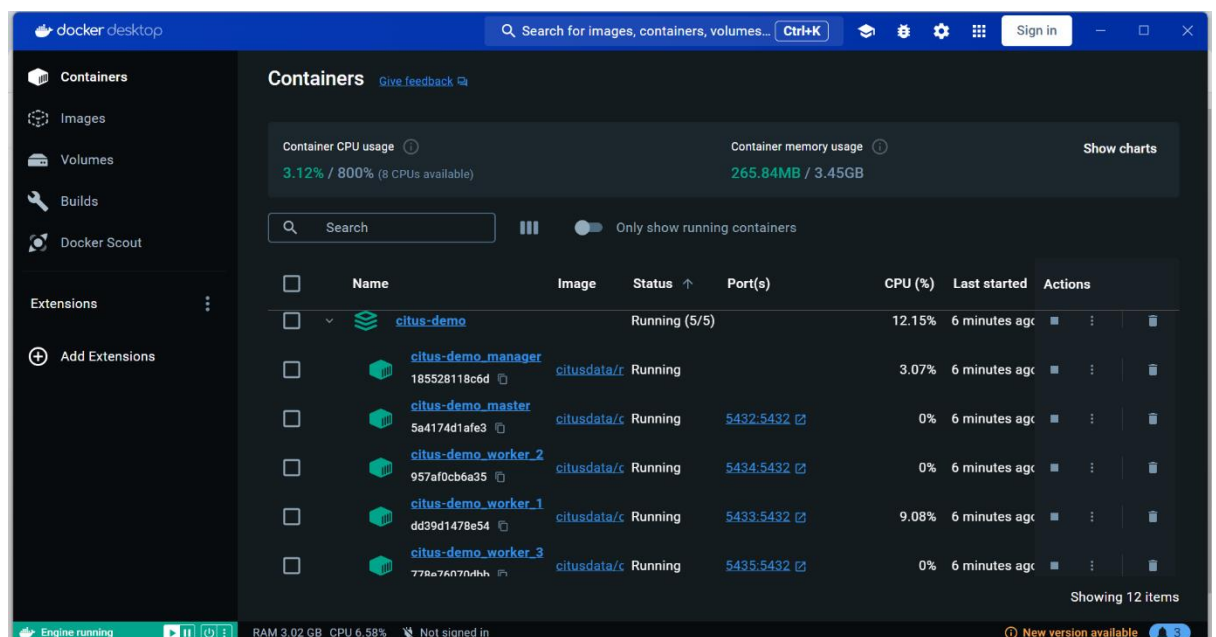
Jawab :

Menjalankan Citus menggunakan Docker Compose

`docker compose up -d`



```
root@LAPTOP-R9NVHPVL:~/citus-demo# docker compose up -d
WARN[0000] /root/citus-demo/docker-compose.yml: `version` is obsolete
[+] Running 6/6
 ✓ Network citus-demo_postgres-network Created 0.1s
 ✓ Container citus-demo_master Started 2.5s
 ✓ Container citus-demo_manager Started 3.0s
 ✓ Container citus-demo_worker_3 Started 4.9s
 ✓ Container citus-demo_worker_1 Started 4.6s
 ✓ Container citus-demo_worker_2 Started 4.8s
root@LAPTOP-R9NVHPVL:~/citus-demo#
```



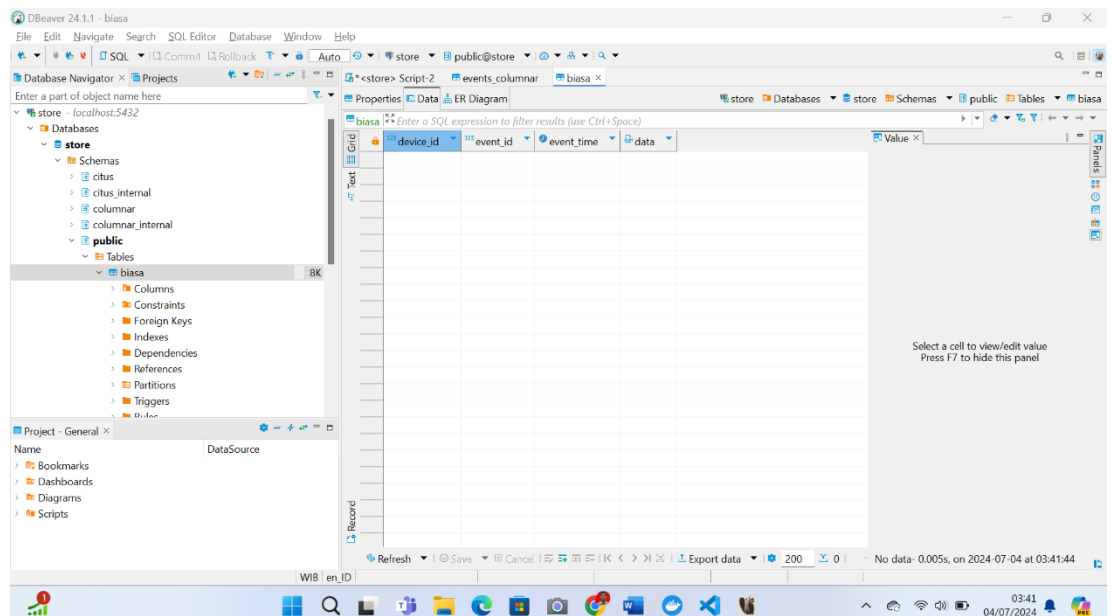
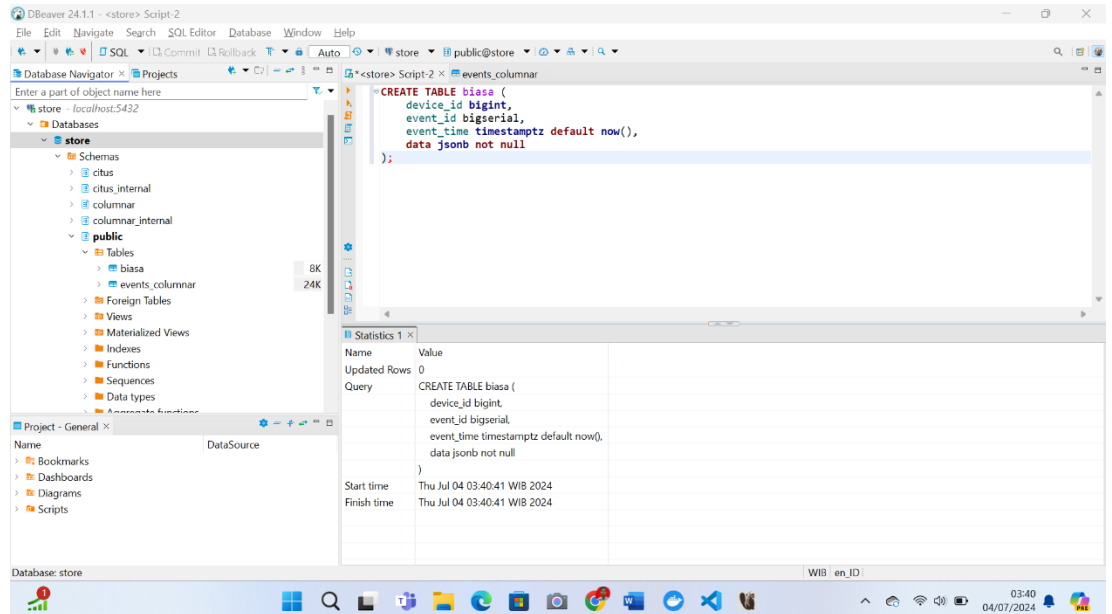
2. Tuliskan perintah untuk membuat

a. Tabel Biasa

Jawab :

Perintah untuk membuat tabel biasa

```
CREATE TABLE biasa (  
    device_id bigint,  
    event_id bigserial,  
    event_time timestamptz default now(),  
    data jsonb not null  
);
```

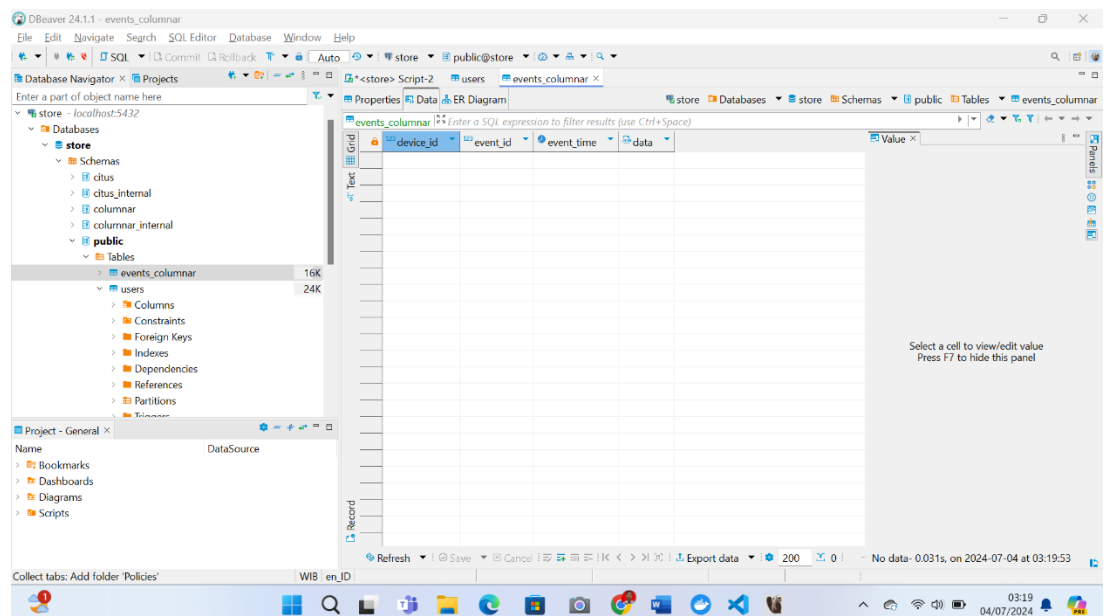
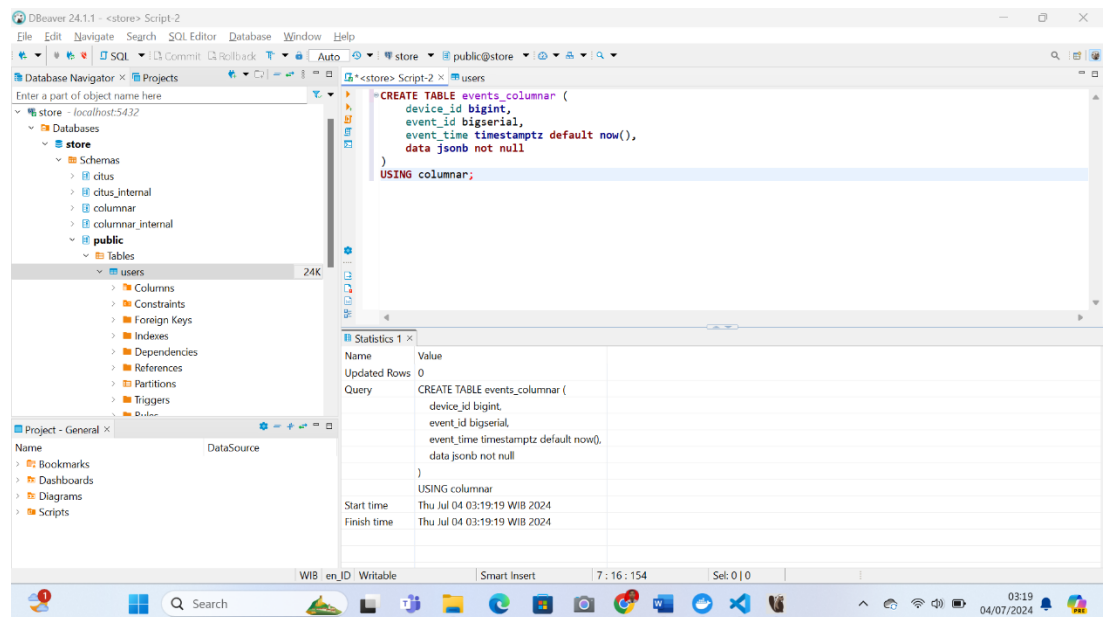


b. Columnar tabel

Jawab :

Perintah untuk membuat tabel columnar

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



3. Masukkan 1000 baris data ke dalam tabel biasa dan tabel columnar

a. Tabel Biasa

Jawab :

```
INSERT INTO biasa (device_id, data)  
SELECT d, '{"hello":"nando"}' FROM generate_series(1,1000) d;
```

The screenshot shows the DBeaver 24.1.1 interface. The SQL Editor on the right contains the following SQL script:

```
CREATE TABLE biasa (  
    device_id bigint,  
    event_id bigint,  
    event_time timestampz default now(),  
    data jsonb not null  
);  
  
INSERT INTO biasa (device_id, data)  
SELECT d, '{"hello":"nando"}' FROM generate_series(1,1000) d;
```

The Database Navigator on the left shows the schema structure. The Statistics window at the bottom right displays the following information:

Name	Value
Updated Rows	1000
Query	INSERT INTO <i>biasa</i> (<i>device_id</i> , <i>data</i>) SELECT <i>d</i> , '{"hello":"nando"}' FROM generate_series(1,1000) <i>d</i>
Start time	Thu Jul 04 03:42:56 WIB 2024
Finish time	Thu Jul 04 03:42:56 WIB 2024

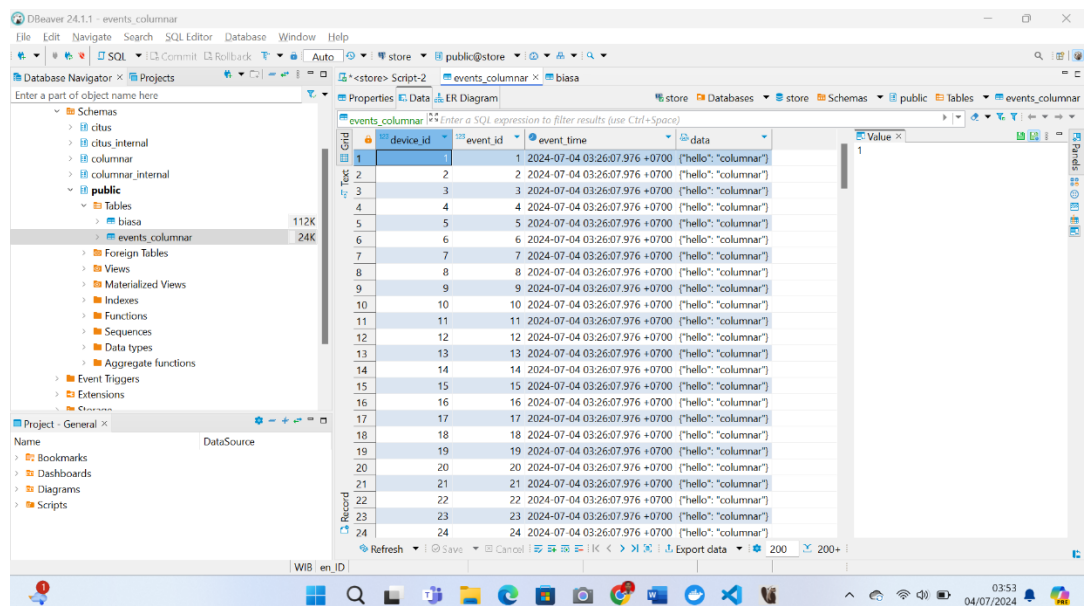
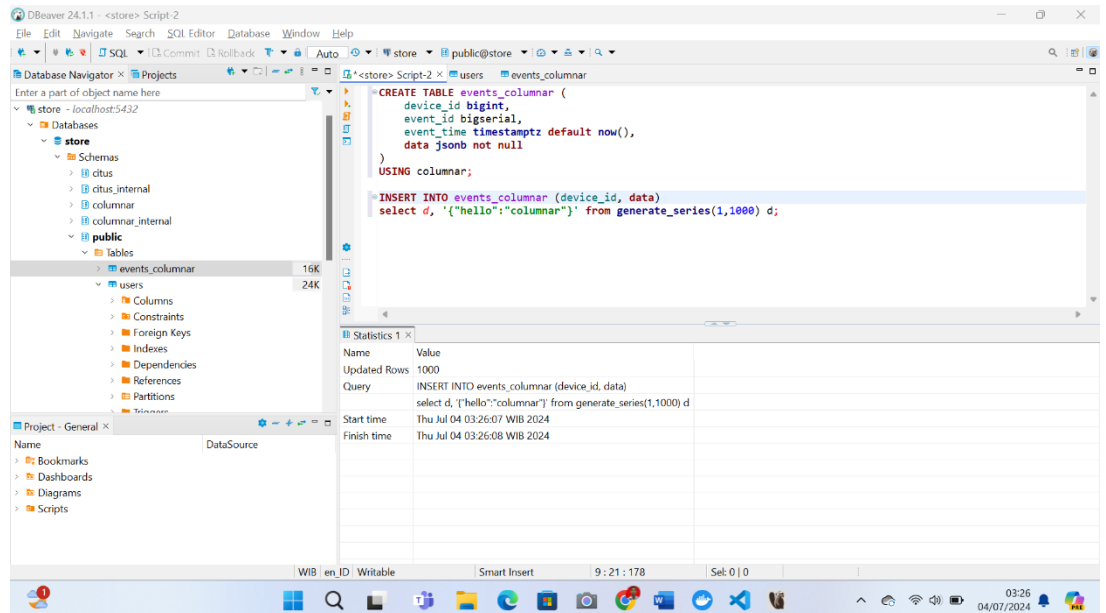
The screenshot shows the DBeaver 24.1.1 interface with the Data view of the 'biasa' table. The table has the following columns: device_id, event_id, event_time, and data. The data is displayed in a grid view, showing the first 24 rows of the 1000 inserted records. The 'data' column contains the JSON string '{"hello":"nando"}' for all rows.

Grid	device_id	event_id	event_time	data
1	1	1	2024-07-04 03:42:56.251 +0700	['hello': 'nando']
2	2	2	2024-07-04 03:42:56.251 +0700	['hello': 'nando']
3	3	3	2024-07-04 03:42:56.251 +0700	['hello': 'nando']
4	4	4	2024-07-04 03:42:56.251 +0700	['hello': 'nando']
5	5	5	2024-07-04 03:42:56.251 +0700	['hello': 'nando']
6	6	6	2024-07-04 03:42:56.251 +0700	['hello': 'nando']
7	7	7	2024-07-04 03:42:56.251 +0700	['hello': 'nando']
8	8	8	2024-07-04 03:42:56.251 +0700	['hello': 'nando']
9	9	9	2024-07-04 03:42:56.251 +0700	['hello': 'nando']
10	10	10	2024-07-04 03:42:56.251 +0700	['hello': 'nando']
11	11	11	2024-07-04 03:42:56.251 +0700	['hello': 'nando']
12	12	12	2024-07-04 03:42:56.251 +0700	['hello': 'nando']
13	13	13	2024-07-04 03:42:56.251 +0700	['hello': 'nando']
14	14	14	2024-07-04 03:42:56.251 +0700	['hello': 'nando']
15	15	15	2024-07-04 03:42:56.251 +0700	['hello': 'nando']
16	16	16	2024-07-04 03:42:56.251 +0700	['hello': 'nando']
17	17	17	2024-07-04 03:42:56.251 +0700	['hello': 'nando']
18	18	18	2024-07-04 03:42:56.251 +0700	['hello': 'nando']
19	19	19	2024-07-04 03:42:56.251 +0700	['hello': 'nando']
20	20	20	2024-07-04 03:42:56.251 +0700	['hello': 'nando']
21	21	21	2024-07-04 03:42:56.251 +0700	['hello': 'nando']
22	22	22	2024-07-04 03:42:56.251 +0700	['hello': 'nando']
23	23	23	2024-07-04 03:42:56.251 +0700	['hello': 'nando']
24	24	24	2024-07-04 03:42:56.251 +0700	['hello': 'nando']

b. Tabel Columnar

Jawab :

```
INSERT INTO events_columnar (device_id, data)
SELECT d, '{"hello":"columnar"}' FROM generate_series(1,1000) d;
```



4. Tampilkan perbedaan ukuran antara tabel biasa dan tabel columnar

Jawab : Berikut tampilan perbedaan tabel biasa dan tabel columnar

public	
Tables	
biasa	112K
events_columnar	24K

5. Tuliskan kesimpulannya

Jawab : Kesimpulannya ialah tabel biasa menyimpan data berdasarkan baris dan penyimpanan memorinya lebih besar dibanding tabel columnar sedangkan tabel columnar menyimpan data berdasarkan kolom dan kompresi datanya lebih baik dibanding tabel biasa