

PART 1 – Introduction to Data Warehouse

TASK

1. Sebutkan perbedaan antara data warehouse dan data lake

Jawabannya : Berikut adalah perbedaan antara data warehouse dan data lake:

➤ Definisi :

Data warehouse adalah kumpulan data yang disusun dan diintegrasikan dari berbagai sumber yang berbeda untuk tujuan analisis bisnis. Data warehouse dirancang untuk menyimpan data yang telah diproses, terstruktur, dan dioptimalkan untuk pelaporan dan analisis bisnis.

Data lake adalah penyimpanan yang fleksibel dan skema-agnostic untuk data mentah, semi-terstruktur, dan tidak terstruktur. Data lake menyediakan tempat untuk menyimpan semua jenis data tanpa perlu mengubah skema data secara sebelumnya.

➤ Struktur Data:

Data Warehouse: Biasanya menyimpan data yang sudah diproses, terstruktur, dan diatur dalam format yang telah ditentukan sebelumnya. Data warehouse menggunakan skema yang telah dirancang sebelumnya dan biasanya cocok untuk analisis bisnis.

Data Lake: Menyimpan data mentah (raw data) dan juga data yang sudah diproses. Data lake tidak memerlukan skema yang ketat dan memungkinkan penyimpanan berbagai jenis data dalam format yang asli.

➤ Tujuan Utama:

Data Warehouse: Dirancang untuk analisis data dan pelaporan bisnis. Fokusnya pada konsistensi, keandalan, dan ketersediaan data yang tinggi.

Data Lake: Dibuat untuk menyimpan semua jenis data, baik terstruktur maupun tidak terstruktur, dengan tujuan memfasilitasi eksplorasi dan analisis data yang lebih luas. Dapat digunakan untuk analisis mendalam dan aplikasi kecerdasan buatan.

➤ Skema dan Schema-on-Read:

Data Warehouse: Menggunakan skema berbasis bintang (star schema) atau skema salju (snowflake schema) untuk menyusun data secara terstruktur sebelum disimpan. Data diintegrasikan dan disiapkan sebelum diakses.

Data Lake: Menggunakan pendekatan schema-on-read, yang berarti data disimpan tanpa perlu diubah menjadi format tertentu terlebih dahulu. Skema diterapkan saat data dibaca atau saat data dianalisis.

➤ Sifat dan Fleksibilitas:

Data Warehouse: Lebih terstruktur, lebih terkendali, dan cenderung memiliki latensi rendah dalam pengambilan data. Cocok untuk aplikasi yang membutuhkan konsistensi dan kecepatan akses yang tinggi.

Data Lake : Lebih fleksibel dalam hal jenis dan volume data yang dapat disimpan. Cocok untuk pengolahan big data dan eksplorasi data yang membutuhkan skala besar dan variasi data yang tinggi.

➤ Kesesuaian Penggunaan:

Data Warehouse: Ideal untuk analisis bisnis, pelaporan, dan kebutuhan analisis yang terstruktur dan terencana.

Data Lake: Cocok untuk eksplorasi data, analisis mendalam, dan aplikasi kecerdasan buatan yang membutuhkan akses ke data mentah dan beragam.

Dengan memahami perbedaan ini, kita dapat memilih antara data warehouse dan data lake berdasarkan kebutuhan spesifik proyek atau organisasi.

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

Jawabannya : Perbedaan antara teknologi database untuk data warehouse (OLAP) dan teknologi database konvensional (OLTP) dapat dijelaskan dari beberapa aspek utama:

OLTP (Online Transaction Processing)

Tujuan Utama:

- **Operasional:** Didesain untuk mendukung transaksi harian dan operasional dalam bisnis, seperti entri pesanan, pembaruan inventaris, dan transaksi harian lainnya.

Karakteristik Utama:

- **Jumlah Transaksi Tinggi:** Memproses banyak transaksi pendek dan cepat.
- **Normalisasi:** Menggunakan skema normalisasi untuk menghindari redundansi data dan memastikan konsistensi data.
- **Optimasi Write:** Fokus pada operasi menulis (INSERT, UPDATE, DELETE).
- **Ketersediaan dan Konsistensi:** Menjamin ketersediaan data real-time dan konsistensi transaksi.
- **Penggunaan Indeks:** Menerapkan indeks untuk meningkatkan kinerja pencarian data.

Contoh Sistem: MySQL, PostgreSQL, Oracle Database, SQL Server.

OLAP (Online Analytical Processing)

Tujuan Utama:

- **Analitis:** Dibuat untuk menganalisis data yang besar dan kompleks untuk mendukung pengambilan keputusan strategis.

Karakteristik Utama:

- **Aggregasi Data:** Menggunakan data agregat untuk analisis yang luas.
- **Denormalisasi:** Menciptakan skema yang denormalisasi untuk memfasilitasi kueri analitis yang kompleks dan cepat.
- **Optimasi Read:** Fokus pada operasi membaca data (SELECT).
- **Kompleksitas Kueri:** Mendukung kueri kompleks dan analisis multidimensional.
- **Penyimpanan Data Lama:** Memungkinkan penyimpanan data sejarah untuk analisis tren jangka panjang.
- **Penggunaan Cube/Data Warehouse:** Menggunakan data warehouse atau cube untuk menyimpan dan mengelola data analitis.

Contoh Sistem: Apache Hive, Apache Hadoop, Vertica, Teradata.

Perbandingan Singkat

Tujuan Utama: OLTP digunakan untuk transaksi harian dan operasional, sedangkan OLAP digunakan untuk analisis dan pelaporan data.

Struktur Data: OLTP menggunakan skema normalisasi untuk menghindari redundansi, sementara OLAP menggunakan skema denormalisasi untuk kueri analitis yang cepat.

Operasi Utama: OLTP mengoptimalkan operasi menulis dan memastikan konsistensi dan ketersediaan data real-time, sementara OLAP mengoptimalkan operasi membaca untuk mendukung kueri analitis yang kompleks.

Penggunaan Indeks: OLTP menggunakan indeks untuk meningkatkan kinerja pencarian data dalam transaksi, sedangkan OLAP mungkin menggunakan indeks tapi lebih fokus pada struktur data dan pengolahan kueri.

Dengan memahami perbedaan ini, organisasi dapat memilih teknologi database yang sesuai dengan kebutuhan operasional atau analitis mereka, memastikan efisiensi dan kinerja yang optimal sesuai dengan tujuan penggunaan data.

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

Jawabannya : Ada beberapa teknologi utama yang biasanya dipakai untuk membangun dan mengelola data warehouse. Berikut adalah beberapa di antaranya:

I. Relational Database Management Systems (RDBMS):

- **Contoh:** Oracle Database, SQL Server, MySQL, PostgreSQL
- **Deskripsi:** RDBMS masih menjadi pilihan utama untuk data warehouse karena memiliki kemampuan untuk menyimpan data terstruktur dalam skema yang sudah dirancang sebelumnya (seperti skema bintang atau skema salju). Mereka juga menyediakan dukungan untuk bahasa SQL yang kuat untuk melakukan kueri dan analisis data.

II. Columnar Database Systems:

- **Contoh:** Amazon Redshift, Google BigQuery, Vertica, Snowflake
- **Deskripsi:** Database sistem kolom mengorganisasi data dalam kolom daripada baris, yang membuatnya sangat efisien untuk kueri yang melibatkan agregasi dan analisis data yang besar. Mereka cocok untuk beban kerja analitis skala besar dan mendukung kueri kompleks dengan kinerja yang tinggi.

III. Data Warehouse Appliances:

- **Contoh:** IBM Netezza, Teradata
- **Deskripsi:** Data warehouse appliances adalah solusi khusus yang dirancang untuk kinerja tinggi dan skalabilitas dalam pengolahan data analitis. Mereka biasanya terintegrasi dengan perangkat keras dan perangkat lunak yang dioptimalkan untuk analisis data yang cepat dan efisien.

IV. Hadoop Ecosystem:

- **Contoh:** Apache Hive, Apache Hadoop (HDFS), Spark SQL
- **Deskripsi:** Hadoop dan ekosistemnya menawarkan solusi untuk menyimpan dan mengelola data besar (big data) dalam lingkungan terdistribusi. Apache Hive, misalnya, memungkinkan untuk melakukan kueri SQL terhadap data yang disimpan di Hadoop Distributed File System (HDFS).

V. Cloud-based Data Warehouses:

- **Contoh:** Amazon Redshift, Google BigQuery, Snowflake, Azure Synapse Analytics
- **Deskripsi:** Layanan data warehouse berbasis cloud yang menawarkan skalabilitas yang tinggi, performa yang baik, dan fleksibilitas dalam hal manajemen data dan biaya. Mereka memungkinkan organisasi untuk menyimpan, mengelola, dan menganalisis data secara efisien tanpa harus mengelola infrastruktur fisik.

VI. NoSQL Databases (untuk skenario khusus):

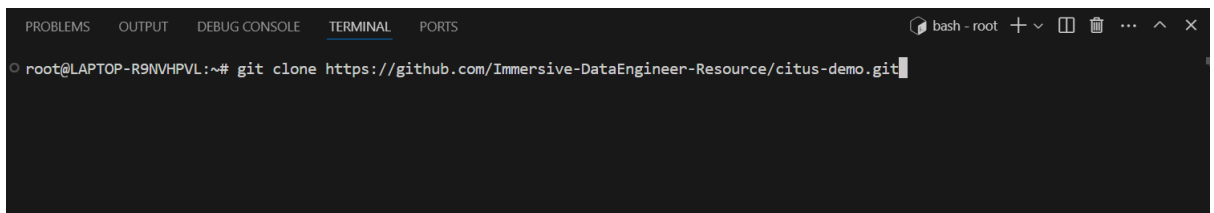
- **Contoh:** Cassandra, MongoDB
- **Deskripsi:** Dalam beberapa kasus, basis data NoSQL dapat digunakan untuk skenario data warehouse yang membutuhkan fleksibilitas dalam menangani data semi-terstruktur dan tidak terstruktur, meskipun umumnya lebih umum digunakan untuk aplikasi operasional (OLTP).

Pilihan teknologi data warehouse akan tergantung pada kebutuhan spesifik organisasi, skala data yang dihadapi, dan jenis analisis yang dilakukan. Semakin banyak perusahaan beralih ke solusi data warehouse yang terkelola secara cloud untuk mendapatkan keuntungan dalam hal fleksibilitas, skalabilitas, dan efisiensi biaya.

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

Jawabannya : berikut langkah-langkah perintah instalasi citus menggunakan docker compose sampai tabel terbentuk!

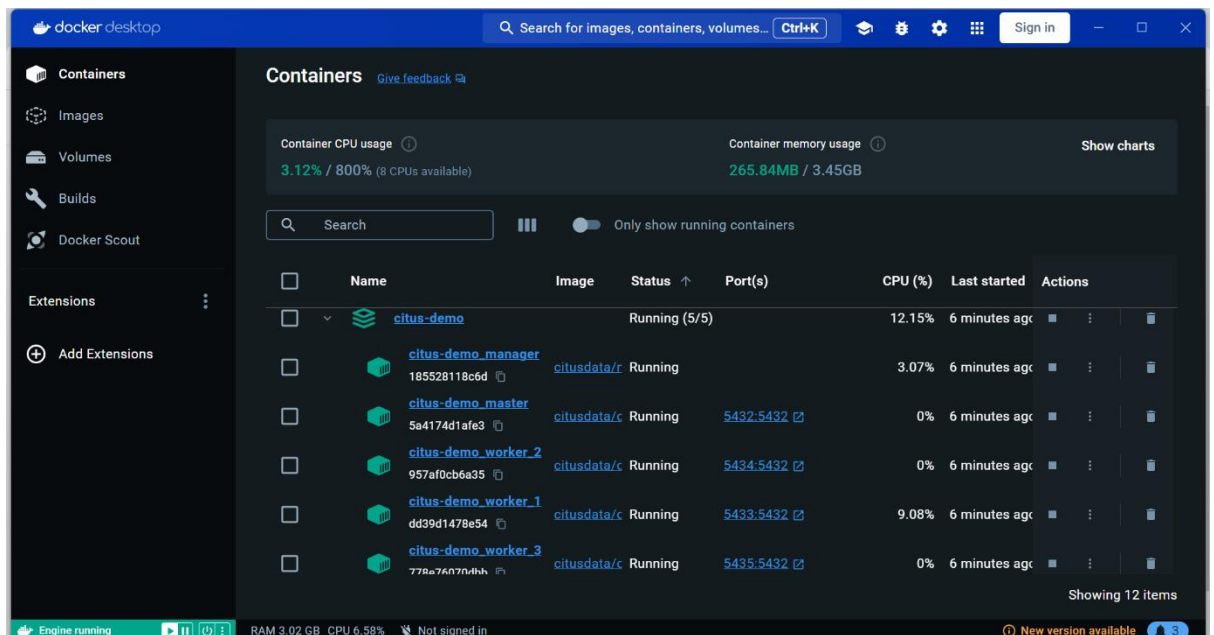
- Mengclone folder citus-demo yang sudah disediakan beserta file docker-compose.yml untuk mengatur layanan citus dan DBeaver**



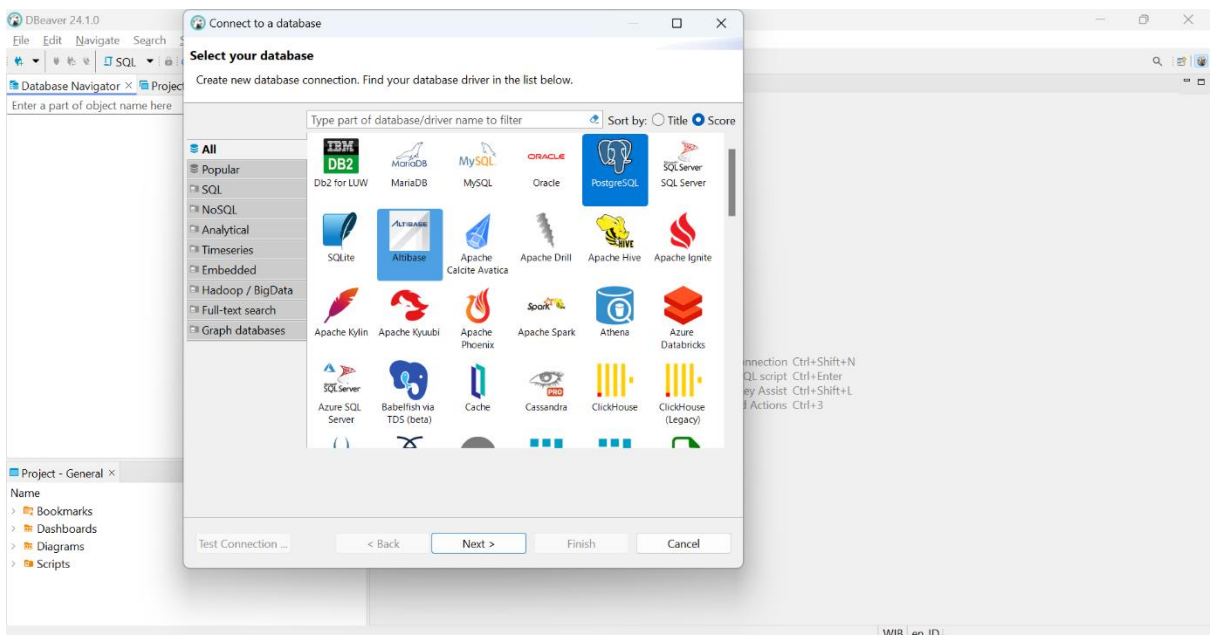
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
root@LAPTOP-R9NVHPVL:~# git clone https://github.com/Immersive-DataEngineer-Resource/citus-demo.git
```

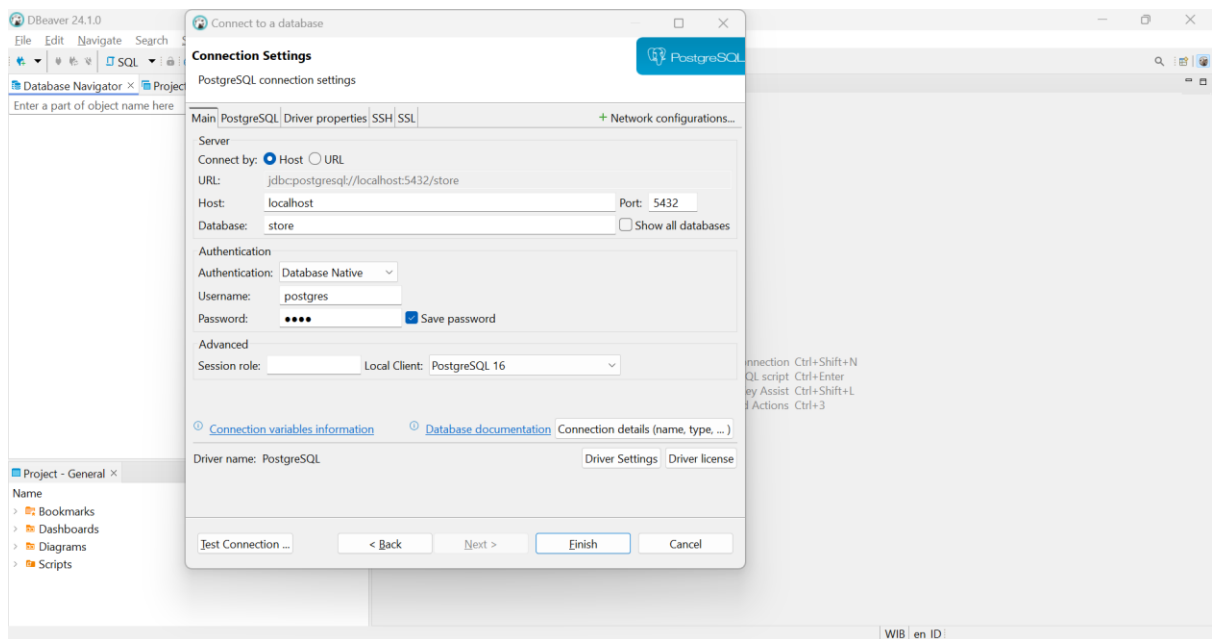
- Jalankan docker compose dengan perintah “docker-compose up -d” ini akan mendownload image citus dari Docker Hub jika belum ada, dan memulai layanan Citus (koordinator dan pekerja) serta DBeaver didalam container Docker.**

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
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#
```

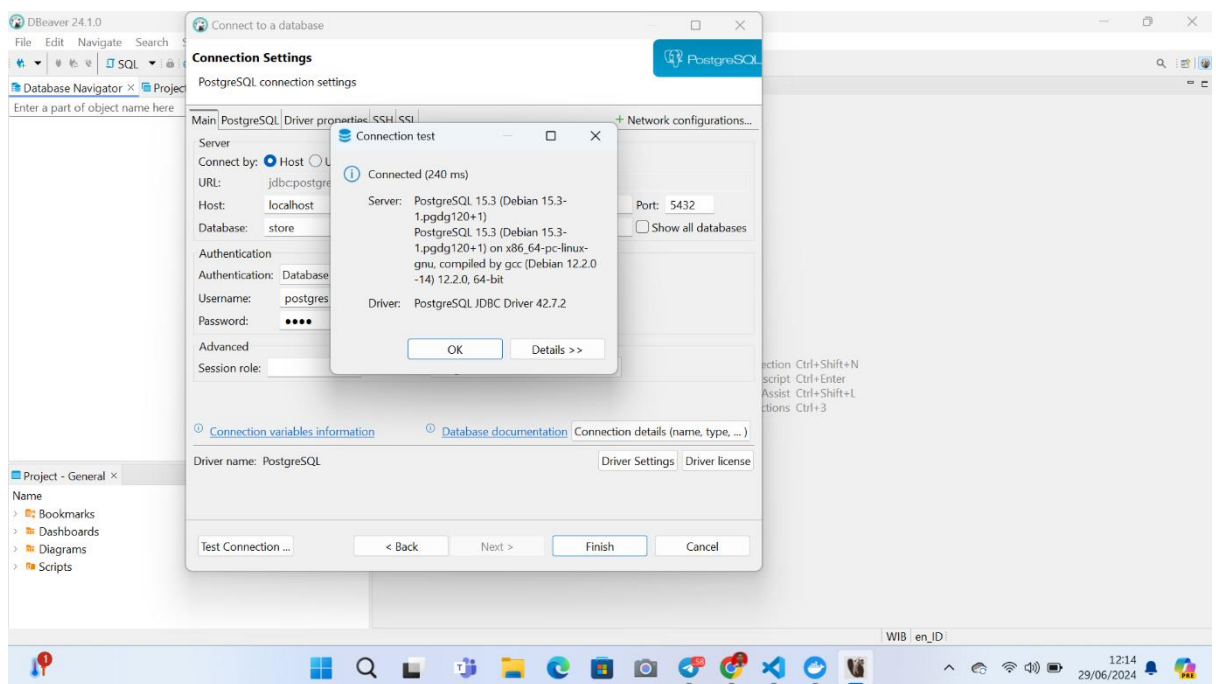


iii. Setelah itu kita buat koneksi dari docker compose

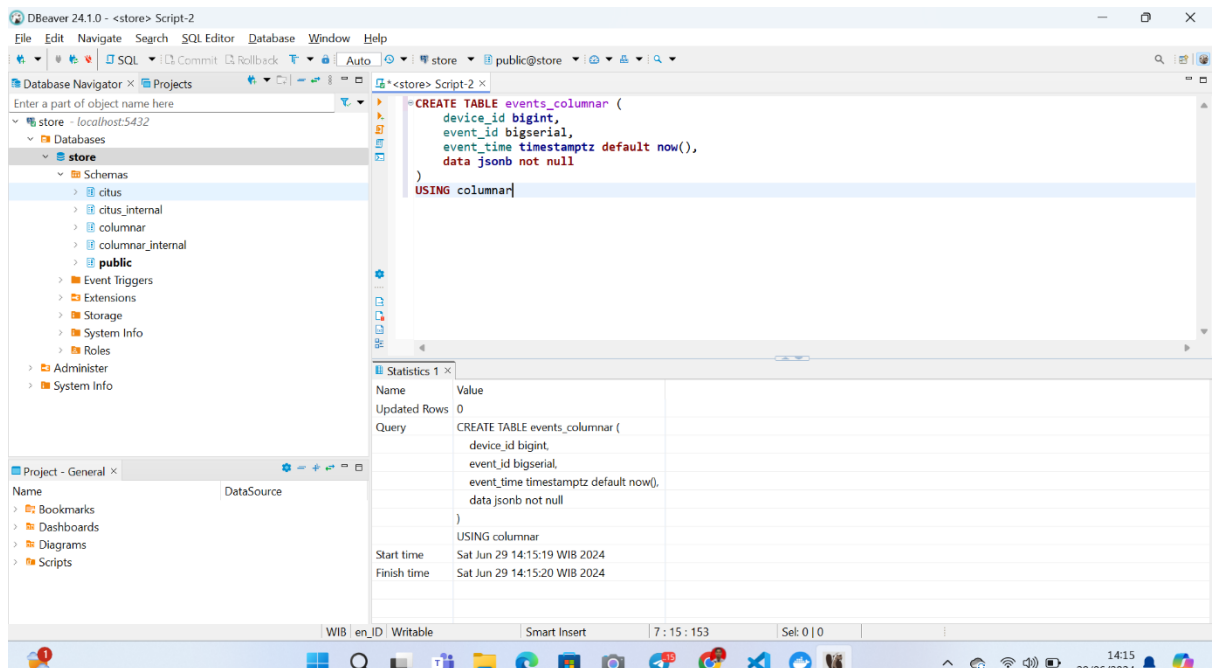




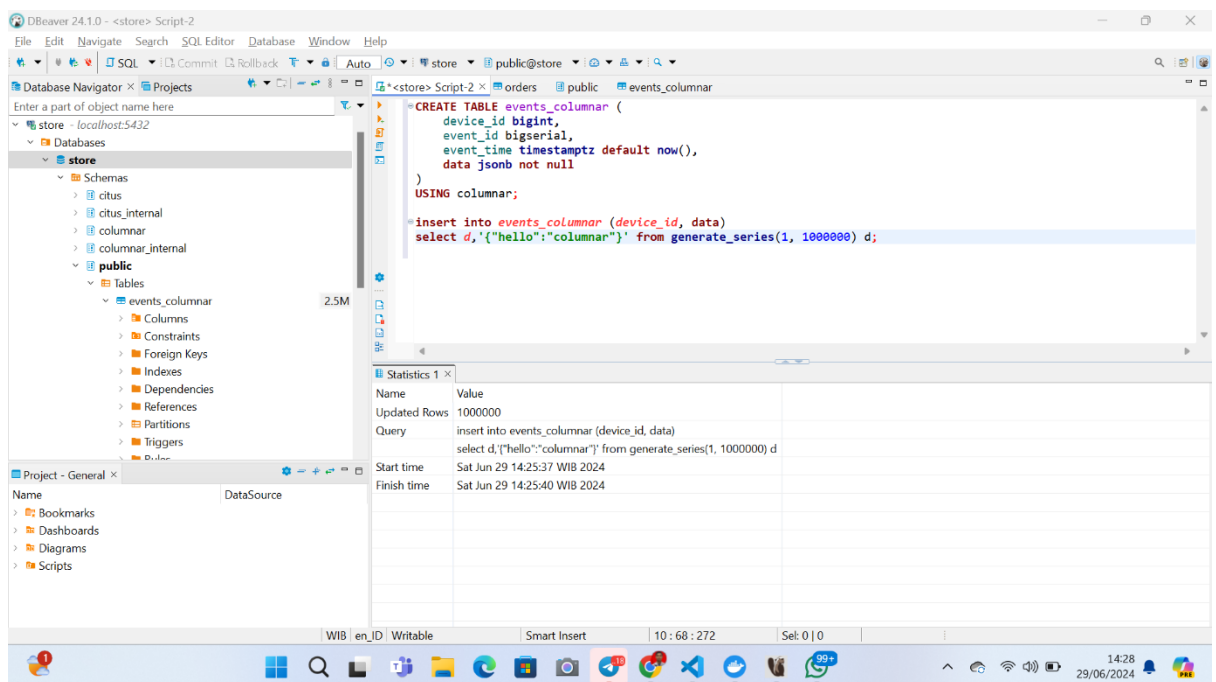
iv. Setelah itu kita test connection



v. Kemudian kita akan membuat tabel menggunakan tabel columnar



vi. Setelah itu kita insert beberapa data ke dalam tabel events_columnar



vii. Berikut tampilan isi datanya

device_id	event_id	event_time	data
1	1	2024-06-29 14:25:37.830 +0700	'hello': 'columnar'
2	2	2024-06-29 14:25:37.830 +0700	'hello': 'columnar'
3	3	2024-06-29 14:25:37.830 +0700	'hello': 'columnar'
4	4	2024-06-29 14:25:37.830 +0700	'hello': 'columnar'
5	5	2024-06-29 14:25:37.830 +0700	'hello': 'columnar'
6	6	2024-06-29 14:25:37.830 +0700	'hello': 'columnar'
7	7	2024-06-29 14:25:37.830 +0700	'hello': 'columnar'
8	8	2024-06-29 14:25:37.830 +0700	'hello': 'columnar'
9	9	2024-06-29 14:25:37.830 +0700	'hello': 'columnar'
10	10	2024-06-29 14:25:37.830 +0700	'hello': 'columnar'
11	11	2024-06-29 14:25:37.830 +0700	'hello': 'columnar'
12	12	2024-06-29 14:25:37.830 +0700	'hello': 'columnar'
13	13	2024-06-29 14:25:37.830 +0700	'hello': 'columnar'
14	14	2024-06-29 14:25:37.830 +0700	'hello': 'columnar'
15	15	2024-06-29 14:25:37.830 +0700	'hello': 'columnar'
16	16	2024-06-29 14:25:37.830 +0700	'hello': 'columnar'
17	17	2024-06-29 14:25:37.830 +0700	'hello': 'columnar'
18	18	2024-06-29 14:25:37.830 +0700	'hello': 'columnar'
19	19	2024-06-29 14:25:37.830 +0700	'hello': 'columnar'
20	20	2024-06-29 14:25:37.830 +0700	'hello': 'columnar'
21	21	2024-06-29 14:25:37.830 +0700	'hello': 'columnar'
22	22	2024-06-29 14:25:37.830 +0700	'hello': 'columnar'
23	23	2024-06-29 14:25:37.830 +0700	'hello': 'columnar'

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

Jawabannya : Perbedaan antara access method heap (atau disebut juga sebagai table heap) dan columnar pada Citus berkaitan dengan cara penyimpanan dan pengaturan data di dalam tabel. Berikut adalah penjelasan singkat tentang masing-masing metode:

✓ **Access Method Heap (Table Heap)**

Penyimpanan Data:

- **Heap Storage:** Dalam metode akses heap, data disimpan dalam format baris yang berurutan. Ini berarti setiap baris dari tabel disimpan satu per satu secara berurutan di blok penyimpanan.

Karakteristik Utama:

- **Struktur Penyimpanan:** Data disimpan dalam bentuk yang mirip dengan penyimpanan pada database relasional tradisional. Setiap baris berisi semua kolom yang ada pada tabel.
- **Akses Data:** Cocok untuk transaksi dan operasi OLTP (Online Transaction Processing) yang memerlukan akses cepat ke data dalam jumlah kecil atau sedang.
- **Keuntungan:** Memiliki kinerja yang baik untuk operasi transaksi, karena pengambilan data berdasarkan baris dan lebih cocok untuk aplikasi yang membutuhkan perubahan data yang sering.

Penggunaan pada Citus:

- Access method heap pada Citus digunakan untuk tabel-tabel yang membutuhkan operasi transaksi dan akses data yang sering mengubah data dalam jumlah kecil atau sedang, seperti data referensi atau metadata yang perlu dimodifikasi secara dinamis.

✓ **Columnar Storage**

Penyimpanan Data:

- **Columnar Storage:** Dalam metode penyimpanan columnar, data disimpan dalam kolom-kolom yang terpisah. Artinya, setiap kolom dari tabel disimpan dalam blok penyimpanan yang berbeda secara terpisah.

Karakteristik Utama:

- **Struktur Penyimpanan:** Setiap kolom disimpan secara terpisah, memungkinkan kueri untuk hanya mengakses kolom yang diperlukan tanpa perlu membaca semua kolom dalam satu baris.
- **Akses Data:** Cocok untuk analisis data dan operasi OLAP (Online Analytical Processing) yang memerlukan kueri analitis yang kompleks, seperti agregasi, penggabungan, dan analisis berdasarkan kolom-kolom tertentu.
- **Keuntungan:** Memberikan kinerja yang lebih baik untuk kueri analitis yang melibatkan banyak baris tetapi hanya beberapa kolom tertentu, karena hanya perlu membaca data dari kolom yang relevan.

Penggunaan pada Citus:

- Columnar storage pada Citus digunakan untuk tabel-tabel yang digunakan dalam analisis data skala besar, di mana kueri analitis seperti penggabungan data atau agregasi dilakukan secara intensif. Ini meningkatkan efisiensi kueri dengan memungkinkan akses langsung ke data kolom tanpa perlu memuat semua data baris.

Kesimpulan

Perbedaan utama antara access method heap dan columnar pada Citus terletak pada cara penyimpanan dan pengaturan data di dalam tabel. Heap storage cocok untuk transaksi dan akses data yang sering mengubah sedangkan columnar storage lebih cocok untuk analisis data yang memerlukan kueri analitis kompleks. Pemilihan metode penyimpanan ini harus disesuaikan dengan kebutuhan aplikasi dan jenis operasi data yang akan dilakukan untuk mencapai kinerja yang optimal.