Rakamin Academy

id/x partners

Project Based Internship

Data Warehouse Partitioning

Principles of Partitioning

Daftar Isi

Exploration Source

[Article]

- What is the best way to partition data in a data warehouse integration project?
- MySQL Partitioning :: 3 Partitioning Types
- Data Partitioning: What It Is and Why It Matters | Timescale

[Video]

- Partitioning in Data Warehouse | Lecture #12 | Data Warehouse Tutorial

A. Introduction

Dalam dunia *Financial Technology* (*fintech*) atau *Banking*, data menjadi hal yang paling penting dengan segala kompleksitasnya. Setiap hari akan ada data yang mengalir, sehingga sangat dibutuhkan bagi seorang Data Engineer yang ingin berkarir di dunia *fintech* agar dapat memahami strategi *data partitioning*. Data partitioning merujuk pada teknik penyusunan dan pemisahan data menjadi subset yang lebih kecil (disebut partisi) berdasarkan kriteria tertentu. Penerapan partisi dalam *data warehouse* dirancang untuk meningkatkan efisiensi, kinerja, dan manajemen data secara keseluruhan, terutama dalam konteks analisis dan pemrosesan data skala besar. Tujuan utamanya itu sendiri adalah memungkinkan pemrosesan data yang lebih cepat, penyimpanan yang lebih efisien, dan optimasi query, terutama ketika hanya sebagian kecil data yang relevan untuk query atau analisis yang diinginkan.

Pada artikel ini, kita akan mengeksplorasi prinsip-prinsip *data warehouse partitioning*, dan berusaha untuk memposisikan diri sebagai seorang Data Engineer dengan memahami konsep menentukan *key partition* yang tepat, dan cara membuat *partition* pada *data warehouse* menggunakan SQL Query.

B. Principles of Partitioning

Perlu diingat bahwa tidak semua data membutuhkan *partitioning*. Penerapan partisi dalam *data warehouse* atau *database* dapat dipertimbangkan dalam beberapa situasi dan kebutuhan tertentu. Beberapa kondisi di mana menerapkan partisi mungkin sangat bermanfaat meliputi :

1. **Volume Data Besar**

   Ketika ukuran *dataset* sangat besar, partisi dapat membantu meningkatkan kinerja query dengan membatasi pemindaian hanya pada bagian-bagian tertentu dari data.

2. **Pola Querynya Berdasarkan Rentang Waktu**

   Jika seringkali terdapat query atau analisis yang melibatkan rentang waktu tertentu, partisi berdasarkan kolom tanggal atau waktu dapat mempercepat pemilihan data historis.

3. **Akses Data Berdasarkan Kategori**

   Jika data sering diakses atau di query berdasarkan kategori tertentu, seperti jenis transaksi atau area geografis, partisi dapat memfasilitasi akses yang lebih cepat dengan membatasi pemindaian pada partisi yang sesuai.

4. **Manajemen Data yang Efisien**

   Untuk mempermudah operasi manajemen data seperti penghapusan data lama, penambahan data baru, atau pemindahan data antar tabel.

5. **Kriteria Bisnis Khusus**

   Jika terdapat kebutuhan atau kriteria bisnis khusus yang mendukung penggunaan partisi, seperti pemisahan data klien, proyek, atau departemen.

Ketika data memang benar membutuhkan *partitioning*, maka disinilah peran seorang Data Engineer dibutuhkan, yakni bagaimana menentukan *key partitioning* yang tepat dari sebuah data. Sederhananya disini adalah seorang Data Engineer tidak hanya mampu dari sisi teknis saja, namun juga mendalami konsep-konsep non-teknis. Lalu, adakah pendekatan tertentu ketika kita ingin menetapkan *key*

partitioning pada sebuah data? Tentu saja ada. Berikut adalah pendekatan yang perlu dipertimbangkan ketika menentukan *key partitioning* yang tepat dalam data.

1. **Pemahaman Pola Akses Data**

   Identifikasi pola akses data yang umum terjadi. Apakah query atau analisis seringkali melibatkan rentang waktu tertentu, jenis transaksi, atau cabang/lokasi tertentu?

2. **Rentang Waktu Transaksi**

   Jika data seringkali diakses atau di query berdasarkan rentang waktu, pertimbangkan untuk menggunakan kolom tanggal transaksi sebagai kandidat partisi. *Horizontal partitioning* berdasarkan tanggal dapat meningkatkan kinerja query data historis.

3. **Jenis Transaksi atau Kategori Lainnya**

   Jika terdapat jenis transaksi atau kategori lainnya yang sering diakses atau di query secara terpisah, pertimbangkan menggunakan kolom tersebut untuk partisi.

4. **Unik dan Terdistribusi Merata**

   Pilih kolom yang memiliki nilai unik dan terdistribusi merata. Hal ini dapat membantu dalam menghindari ketidakseimbangan partisi.

5. **Kriteria Bisnis**

   Kriteria bisnis berperan krusial dalam penentuan *key partitioning*, mencerminkan kebutuhan dan kebijakan yang dapat berpengaruh pada struktur *data warehouse*. Sebagai contoh, jika retensi data didasarkan pada kategori transaksi seperti "Reguler" dan "Investasi," menggunakan kolom kategori sebagai *key partitioning* mendukung kepatuhan terhadap aturan penyimpanan. Kriteria bisnis juga dapat mencakup aspek keamanan, memungkinkan desain partisi yang memisahkan data sensitif ke dalam

partisi terenkripsi, sesuai dengan perbedaan tingkat sensitivitas antar transaksi. Dengan mempertimbangkan kriteria bisnis, *key partitioning* dapat secara menyeluruh mencerminkan kebutuhan bisnis dan memastikan kepatuhan terhadap standar keamanan serta regulasi yang berlaku.

6. **Kolom yang Sering Di *Join* atau Dicari**

   Jika ada kolom yang sering di-*join* dengan tabel lain atau digunakan sebagai kriteria pencarian dalam query, pertimbangkan menggunakan kolom tersebut untuk partisi.

C. Strategi/Teknik dalam *Data Partitioning*

Secara umum teknik partisi data terdiri atas dua, yakni teknik *Horizontal Partitioning* dan *Vertical Partitioning.* Hal ini akan dibahas lebih detail pada materi selanjutnya, namun sebagai pemahaman diawal perlu untuk diketahui bahwa penerapan *horizontal partitioning* maupun *vertical partitioning* pada data disesuaikan dengan kondisi bisnis dan *behaviour* data di perusahaan.

1. *Horizontal partitioning* → digunakan saat kita ingin memisahkan data berdasarkan **baris atau sejumlah baris tertentu**, efektif untuk mengelompokkan data dengan karakteristik serupa, contoh *simple*-nya adalah membagi (mempartisi) data berdasarkan waktu.
2. *Vertical Partitioning* → memisahkan data berdasarkan **kolom**, memudahkan penyimpanan dan akses yang lebih efisien terhadap sebagian data, contohnya adalah membagi 1 data besar menjadi tabel-tabel detail dengan masing-masing memiliki *key* yang terhubung.

Studi Kasus: Optimalisasi Performa *Data Warehouse* dengan Partisi pada Bank Digital

Sebuah bank digital menghadapi tantangan dalam kinerja *data warehouse* mereka karena pertumbuhan cepat volume data dari transaksi perbankan harian, aplikasi perbankan *online*, dan aktivitas nasabah lainnya. Bank ini memiliki kebutuhan untuk meningkatkan efisiensi query dan manajemen data.

SOLUSI

Untuk mengatasi masalah ini, bank tersebut memutuskan untuk menerapkan partisi pada *data warehouse* mereka. Mereka memilih kolom tanggal transaksi sebagai *key partition* karena seringnya query dan laporan berfokus pada rentang waktu tertentu. Selain itu, mereka mempertimbangkan jenis transaksi sebagai kriteria partisi untuk memisahkan transaksi perbankan utama dari transaksi lainnya.

Hasil dari penerapan partisi ini memberikan dampak positif. Kinerja query meningkat secara signifikan karena hanya partisi yang relevan yang di-*scan* saat eksekusi query, mengurangi beban kerja pada server *database*. Manajemen data historis juga menjadi lebih efisien, dan bank dapat dengan cepat mengakses informasi transaksi tertentu tanpa harus memindai seluruh dataset.

Sebagai hasilnya, bank digital ini berhasil meningkatkan efisiensi operasional, merespon lebih cepat terhadap permintaan analitik, dan mengoptimalkan kinerja *data warehouse* mereka. Penerapan partisi tidak hanya meningkatkan performa,

tetapi juga membantu bank ini untuk membuat keputusan yang lebih baik berdasarkan analisis data yang lebih akurat dan tepat waktu.

REFERENCE

Kimball, R., & Ross, M. (2002). The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling. John Wiley & Sons.

Inmon, W. H., & Hackathorn, R. D. (1994). Using the Data Warehouse. John Wiley & Sons.

Data Warehousing - Partitioning Strategy

MySQL Partitioning :: 3 Partitioning Types