Rakamin Academy

id/x partners

Project Based Internship

Data Warehouse Foundation

Indexes Process

Daftar Isi

Exploration Source

[Article]

- Indexing the Data Warehouse | ITPro Today
- SQL CREATE INDEX Statement

[Video]

- SQL Indexes - Definition, Examples, and Tips
- How do SQL Indexes Work

A. *Introduction*

Dalam dunia *Data Warehouse*, efisiensi query menjadi krusial untuk mendukung pengambilan keputusan yang cepat dan akurat di tingkat perusahaan. Salah satu strategi yang sangat ditekankan adalah penggunaan indeks. Indeks memainkan peran penting dalam mempercepat operasi pencarian dan pemfilteran data, serta memastikan responsivitas yang optimal terhadap analisis yang semakin kompleks. Dalam materi ini, kami akan menjelajahi konsep dasar indexing, strategi pemilihan indeks yang tepat, dan bagaimana implementasi indeks dapat mengubah paradigma kinerja query dalam lingkungan Data Warehouse. Dengan memahami secara mendalam bagaimana indeks berinteraksi dengan query, kita dapat merancang struktur data yang mendukung analisis yang efisien, meningkatkan produktivitas pengguna, dan memberikan nilai tambah yang signifikan bagi operasional bisnis di era informasi yang cepat berubah.

B. Apa itu *Indexing* pada *Data Warehouse*

Kita biasanya berbicara tentang "*indexing*" sebagai suatu strategi yang diimplementasikan pada tabel-tabel di dalam *Data Warehouse* untuk **mempercepat akses data**. Pada umumnya, "*indexes*" di sini adalah indeks pada kolom-kolom tertentu yang sering diakses atau digunakan dalam operasi pencarian.

Sebagai contoh, di dalam *Data Warehouse*, Anda mungkin membuat indeks pada kolom waktu, ID pelanggan, atau kolom lain yang sering digunakan sebagai kriteria pencarian. Dengan cara ini, ketika query dieksekusi, *database* dapat

dengan cepat melihat indeks dan mengurangi jumlah baris yang harus dipindai untuk memenuhi permintaan, sehingga meningkatkan kinerja.

C. Membuat *Indexing*

Indeks memungkinkan pengambilan data yang lebih cepat dari yang disimpan dalam data warehouse. Salah satu aspek penggunaan indeks dalam *data warehouse* yang populer adalah implementasi indeks bitmap. Penggunaan indeks bertujuan untuk meningkatkan kinerja query, mengurangi waktu tanggapan, dan mengoptimalkan penggunaan sumber daya dalam lingkungan *data warehousing*. Penggunaan indeks bitmap secara luas diterapkan dalam lingkungan data warehousing, khususnya pada kasus dengan jumlah data yang besar dan banyak kueri ad hoc, namun tingkat transaksi DML yang rendah.

Teknik indeks ini memberikan beberapa keunggulan, seperti waktu tanggapan yang lebih cepat untuk kueri ad hoc, kebutuhan penyimpanan yang lebih sedikit dibandingkan dengan teknik indeks lainnya, dan peningkatan kinerja yang signifikan, bahkan pada perangkat keras dengan jumlah CPU atau memori yang terbatas. Berbeda dengan indeks B-tree tradisional, melakukan indeks penuh pada tabel besar dengan indeks bitmap lebih ekonomis dalam hal ruang penyimpanan karena indeksnya dapat beberapa kali lebih kecil dari data di dalam tabel.

Berikut adalah contoh SQL query untuk membuat indeks.

1. Membuat indeks pada kolom 'nomor_rekening' di tabel 'nasabah'

```
CREATE INDEX idx_nomor_rekening ON nasabah (nomor_rekening);
```

2. Membuat indeks unik pada kolom 'email' di tabel 'nasabah'

```
CREATE UNIQUE INDEX idx_email ON nasabah (email);
```

3. Membuat indeks pada kolom 'tanggal_transaksi' di tabel 'transaksi'

```
CREATE INDEX idx_tanggal_transaksi ON transaksi (tanggal_transaksi);
```

4. Membuat indeks pada kolom 'kode_cabang' di tabel 'rekening'

```
CREATE INDEX idx_kode_cabang ON rekening (kode_cabang);
```

5. Membuat indeks pada beberapa kolom di tabel 'pinjaman'

```
CREATE INDEX idx_pinjaman_nasabah_tanggal ON pinjaman (id_nasabah, tanggal);
```

6. Membuat indeks pada kolom 'nomor_kartu' di tabel 'kartu_kredit'

```
CREATE INDEX idx_nomor_kartu ON kartu_kredit (nomor_kartu);
```

Penjelasan Query :

1. Index pada Kolom 'nomor_rekening' di Tabel 'nasabah'
   - `CREATE INDEX idx_nomor_rekening ON nasabah (nomor_rekening);`
   - Membuat indeks pada kolom nomor_rekening di tabel nasabah.
2. Index Unik pada Kolom 'email' di Tabel 'nasabah'
   - `CREATE UNIQUE INDEX idx_email ON nasabah (email);`
   - Membuat indeks unik pada kolom email di tabel nasabah.
3. Index pada Kolom 'tanggal_transaksi' di Tabel 'transaksi'
   - `CREATE INDEX idx_tanggal_transaksi ON transaksi (tanggal_transaksi);`
   - Membuat indeks pada kolom tanggal_transaksi di tabel transaksi.

4. Index pada Kolom 'kode_cabang' di Tabel 'rekening'
    - `CREATE INDEX idx_kode_cabang ON rekening (kode_cabang);`
    - Membuat indeks pada kolom kode_cabang di tabel rekening.
5. Index pada Beberapa Kolom di Tabel 'pinjaman'
    - `CREATE INDEX idx_pinjaman_nasabah_tanggal ON pinjaman (id_nasabah, tanggal);`
    - Membuat indeks yang melibatkan lebih dari satu kolom di tabel pinjaman.
6. Index pada Kolom 'nomor_kartu' di Tabel 'kartu_kredit'
    - `CREATE INDEX idx_nomor_kartu ON kartu_kredit (nomor_kartu);`
    - Membuat indeks pada kolom nomor_kartu di tabel kartu_kredit.

Studi Kasus : Optimasi Query dengan Indexing di Sistem Database Bank Syariah

Deskripsi Masalah

Bank Syariah, sebagai lembaga keuangan yang mengelola volume data yang besar, menghadapi kendala dalam kinerja query di lingkungan *Data Warehouse* mereka. Seiring dengan pertumbuhan bisnis dan kompleksitas data yang meningkat, waktu *response query* menjadi semakin lambat. Kondisi ini menghambat efektivitas analisis data, pengambilan keputusan real-time, dan menurunkan produktivitas tim analisis data.

Solusi

1. Identifikasi Query yang Tidak Efisien

Melakukan analisis mendalam terhadap query-query yang sering dieksekusi dan mengidentifikasi query yang memiliki kinerja rendah. Fokus pada query yang melibatkan tabel-tabel besar atau kolom-kolom yang sering digunakan dalam operasi pencarian.

2. Pembuatan Indeks yang Tepat

   Membuat indeks pada kolom-kolom yang sering digunakan dalam klausa WHERE, JOIN, atau ORDER BY dalam query. Misalnya, jika terdapat query pencarian berdasarkan nomor rekening atau tanggal transaksi, membuat indeks pada kolom-kolom tersebut akan meningkatkan kinerja query.

   Contoh pembuatan indeks pada kolom 'nomor_rekening' di tabel 'transaksi'

   CREATE INDEX idx_nomor_rekening ON transaksi (nomor_rekening);

3. Penggunaan Indeks Gabungan (Composite Index)

   Jika query melibatkan beberapa kolom dalam klausa WHERE atau JOIN, pertimbangkan untuk membuat indeks gabungan pada kolom-kolom tersebut. Ini dapat meningkatkan kinerja query yang melibatkan kondisi kompleks.

   Contoh pembuatan indeks gabungan pada kolom 'tanggal' dan 'nomor_rekening' di tabel 'transaksi'

   ```
   CREATE INDEX idx_tanggal_nomor_rekening ON transaksi (tanggal, nomor_rekening);
   ```

4. Pemantauan dan Pemeliharaan Berkala

   Melakukan pemantauan secara berkala terhadap kinerja query dan indeks. Jika terjadi perubahan dalam pola query atau data, melakukan penyesuaian

indeks yang diperlukan. Selain itu, merencanakan pemeliharaan berkala untuk memastikan integritas dan efisiensi indeks.

5. Pengujian Kinerja

   Mengujikan kinerja query setelah implementasi indeks untuk memastikan bahwa perubahan yang dilakukan memberikan dampak positif. Hal ini dapat melibatkan penggunaan alat pengujian kinerja dan analisis eksekusi query sebelum dan sesudah pengindeksan.

Dengan menerapkan strategi ini, Bank Syariah dapat meningkatkan kinerja query, merespons lebih cepat terhadap permintaan analisis data, dan memastikan bahwa sistem database dapat mendukung kebutuhan bisnis secara efisien.

References

How do you optimize data warehouse queries and reports using indexing and partitioning techniques?

Fungsi SQL Index, Contoh, dan Cara Membuatnya 2023

Understanding Data Warehouses: Exploring Data Structures and Indexing Methods