BASIS DATA TERDISTRIBUSI

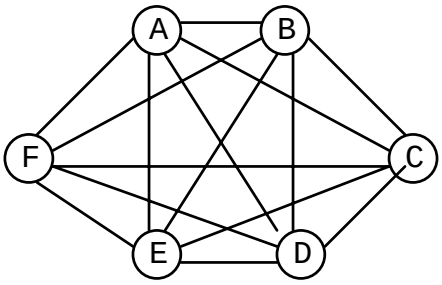
Dalam sebuah database terdistribusi, database disimpan pada beberapa komputer. Komputer-komputer dalam sebuah sistem terdistribusi berhubungan satu sama lain melalui bermacam-macam media komunikasi seperti high-speed buses atau telephone line.

Sebuah sistem database terdistribusi berisikan sekumpulan site, di mana tiap-tiap site dapat berpartisipasi dalam pengeksekusian transaksi-transaksi yang mengakses data pada satu site atau beberapa site. Tiap-tiap site dapat memproses transaksi lokal yaitu sebuah transaksi yang mengakses data pada satu site di mana transaksi telah ditentukan.

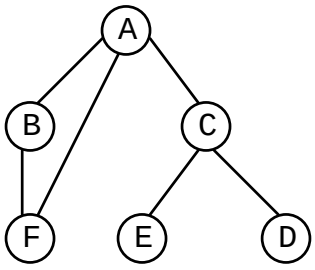
Sebuah site juga dapat mengambil bagian dalam mengeksekusi transaksi global yaitu transaksi yang mengakses data pada site yang berbeda di mana transaksi telah ditentukan, atau transaksi yang mengakses data pada beberapa site yang berbeda.

Untuk menggambarkan kedua tipe transaksi di atas, dapat dimisalkan : transaksi untuk menambahkan $50 pada nomor rekening 177 yang berada di cabang Valleyview. Jika transaksi telah ditentukan pada cabang Valleyview, maka transaksi ini dianggap transaksi lokal. Jika sebuah transaksi untuk mentransfer $50 dari rekening 177 ke rekening 305 yang berlokasi di cabang Hillside, maka transaksi ini dikatakan transaksi global karena rekening di dua site yang berbeda telah diakses sebagai hasil dari eksekusinya.

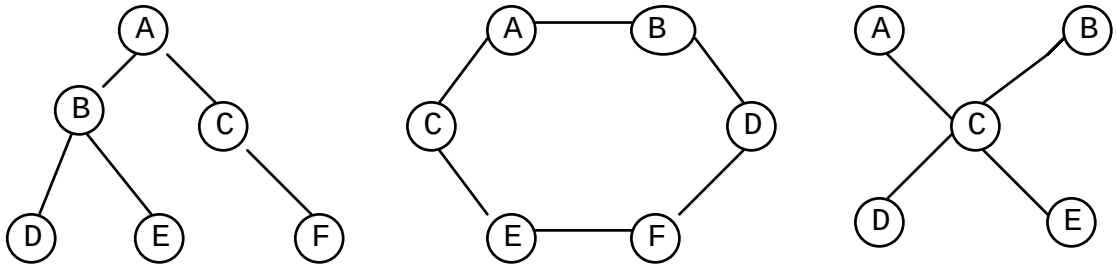
Site-site dalam database terdistribusi dihubungkan secara fisik dengan berbagai cara. Beberapa topologi digambarkan sebagai sebuah graph yang simpul-simpulnya bersesuaian dengan site. Sebuah edge dari simpul A ke simpul B bersesuaian dengan sebuah hubungan langsung antara dua site. Beberapa konfigurasi (bentuk) digambarkan sebagai berikut:



Fully connected network



Partially connected network



Tree structured network Ring network Star network

Gambar 1. Topologi network

Fully Connected network :
Keuntungan : kalau salah satu node rusak, yang lainnya masih dapat berjalan (tetapi biaya mahal).
Kerugian : control management tidak terjamin

Partially connected network :
Keuntungan : reliability rendah, biaya dapat ditekan
Kerugian : control management tidak terjamin

Tree structure network :
Keuntungan : bersifat sentral, control management lebih terjamin
Kerugian : kalau node pusat (A) rusak, semua akan rusak.
Cat : setiap proses dimulai dari bawah.

Ring Network (LAN) :
Keuntungan : rusak satu, yang lain masih berjalan
Kerugian : Control management kurang terjamin karena bersifat desentralisasi

Star Network (LAN) :
Keuntungan : - control management lebih terjamin, karena bersifat sentral
- reliability rendah
Kerugian : kalau pusat rusak, yang lainnya rusak

Keuntungan dan Kerugian Database Terdistribusi

a. Keuntungan-keuntungan dari database terdistribusi

1. Pengawasan distribusi dan pengambilan data
   Jika sejumlah site yang berbeda dihubungkan satu sama lain, lalu seorang pemakai yang berada pada satu site dapat mengakses data yang tersedia pada site lain.
   Sebagai contoh : sistem distribusi pada sebuah bank memungkinkan seorang pemakai pada salah satu cabang dapat mengakses data cabang lain.
2. Reliability dan availability

Sistem distribusi dapat terus menerus berfungsi dalam menghadapi kegagalan dari site individu atau mata rantai komunikasi antar site.
Misal : jika site-site gagal dalam sebuah sistem distribusi, site-site lainnya dapat melanjutkan operasi jika data telah direplikasi pada beberapa site

3. Kecepatan pemrosesan query
   Jika sebuah query melibatkan data pada beberapa site, memungkinkan membagi query ke dalam sub query yang dapat dieksekusi dalam bentuk paralel oleh beberapa site. Perhitungan secara paralel mempercepat pemrosesan dari seorang pemakai query

4. Otonomi lokal
   Pendistribusian sistem mengizinkan sekelompok individu dalam sebuah perusahaan untuk melatih pengawasan lokal melalui data mereka sendiri. Dengan kemampuan ini dapat mengurangi ketergantungan pada pusat pemrosesan.

5. Efisien dan fleksibel
   Data dalam sistem distribusi dapat disimpan dekat dengan titik di mana data tersebut dipergunakan. Data dapat secara dinamik bergerak atau disalin, atau salinannya dapat dihapus.

b. Kerugian-kerugian dari database terdistribusi

1. Harga software yamg mahal
   Hal ini disebabkan sangat sulit untuk membuat sistem database distribusi

2. Kemungkinan kesalahan lebih besar
   Site-site yang termasuk dalam sistem distribusi beroperasi secara paralel sehingga menjadi lebih sulit untuk menjamin kebenaran dari algoritma. Adanya kesalahan mungkin tak dapat diketahui

3. Biaya pemrosesan tinggi
   Perubahan pesan-pesan dan penambahan perhitungan dibutuhkan untuk mencapai koordinasi antar site.

   Dalam memilih sebuah disain untuk sistem database, perancang harus mengimbangi keuntungan dan kerugian dari database terdistribusi.

Fragmentasi Data

Fragmentasi : relasi dipartisikan ke dalam beberapa bagian, setiap bagian disimpan pada lokasi yang berbeda.

Ada beberapa hal yang terlibat dalam penyimpanan relasi pada database terdistribusi di antaranya fragmentasi data. Fragmentasi data memisahkan relasi ke dalam beberapa fragment. Tiap-tiap fragment disimpan pada site yang berbeda.
Pemisahan relasi global ke dalam fragment-fragment dapat disusun dengan menggunakan tiga jenis yang berbeda dari fragmentasi yaitu : *fragmentasi horizontal, fragmentasi vertikal, dan fragmentasi campuran.*

Dalam seluruh jenis fragmentasi, sebuah fragment dapat didefinisikan dengan sebuah ekspresi dalam sebuah bahasa relasional (dalam hal ini digunakan aljabar relasional) yang mengambil relasi global sebagai operan dan memproduksi fragment sebagai hasil.

Beberapa peraturan yang harus diikuti ketika mendefinisikan fragment :

Kondisi lengkap.
Seluruh data dari relasi global harus dipetakan ke dalam fragment. Fragmentasi tidak akan terjadi jika sebuah data item yang dimiliki oleh relasi global, tidak dimiliki oleh beberapa fragment.

Kondisi penyusunan kembali.
Harus selalu mungkin untuk menyusun kembali tiap-tiap relasi global dari fragment-fragmentnya. Hanya fragment-fragment yang disimpan dalam database terdistribusi yang dapat membangun relasi global kembali melalui operasi penyusunan kembali jika diperlukan.

Kondisi disjoin.
Kondisi ini sangat berguna terutama untuk fragmentasi horizontal, sementara untuk fragmentasi vertikal kondisi ini kadang-kadang dilanggar.

Jenis-Jenis Fragmentasi Data

1. *Fragmentasi Horizontal.*
   Fragmentasi horizontal berisikan tuple-tuple yang dipartisi dari sebuah relasi global ke dalam sejumlah subset r_1, r_2, ... , r_n. Tiap-tiap subset berisikan sejumlah tuple dari r. Tiap-tiap tuple dari r harus memiliki satu fragment, sehingga relasi yang asli dapat disusun kembali. Sebuah fragment dalam fragmentasi horizontal dapat didefinisikan sebagai sebuah seleksi pada relasi global r. Oleh karena itu sebuah predikat P_i digunakan untuk menyusun fragment r_i seperti berikut :

   $$r_i = \sigma_i(r)$$

   Penyusunan kembali dari relasi r dapat diperoleh dengan mengambil gabungan dari seluruh fragment :

$$r = \bigcup_{i=1}^{n} r_i$$

2. *Fragmentasi Vertikal*
   Dalam fragmentasi vertikal, tiap-tiap fragment r_i didefinisikan sebagai :

   $$r_i = \pi_i(r)$$

   Relasi global dapat disusun kembali dari fragment-fragment dengan mengambil natural join: $r = r_1 \quad r_2 \quad r \quad ..... \quad r_n$

   Fragmentasi vertikal disempurnakan dengan menambahkan sebuah atribut yang disebut tuple identifier (tuple-id) ke dalam skema r. Sebuah tuple-id adalah sebuah alamat logik dari sebuah tuple. Tiap-tiap tuple dalam r harus memiliki sebuah alamat yang unik, atribut tuple-id sebagai kunci untuk penambahan skema.

   Kunci tersebut akan direplikasikan ke dalam seluruh fragment dengan tujuan untuk penyusunan kembali relasi global. Kita dapat melihat bahwa dalam fragmentasi vertikal motivasi utama untuk memiliki fragment-fragment yang disjoin adalah tidak sepenting dalam fragmentasi horizontal.

3. *Fragmentasi Campuran*
   Relasi r (global) dibagi-bagi ke dalam sejumlah relasi fragment r_1, r_2, r_3, ..., r_n. Tiap-tiap fragment diperoleh sebagai hasil baik dari skema fragmentasi horizontal ataupun skema fragmentasi vertikal pada relasi r, atau dari sebuah fragment r yang diperoleh sebelumnya.

Cara yang sederhana untuk membangun fragmentasi campuran sebagai berikut :
1. Menggunakan fragmentasi horizontal pada fragmentasi vertikal.
2. Menggunakan fragmentasi vertikal pada fragmentasi horizontal.

Contoh : Relasi Deposit

Deposit-Scheme(branch_name, account_number, customer_name, balance)

Branch-name	account-number	Customer-name	balance
Hillside	305	Lowman	500
Hillside	226	Camp	336
Valleyview	177	Camp	205
Valleyview	402	Khan	10000
Hillside	115	Khan	62
Valleyview	408	Khan	1123
Valleyview	639	Green	750

Pemecahan relasi global dengan menggunakan :

1. *Fragmentasi Horizontal :*
   Untuk menggambarkan ini, relasi r adalah relasi deposit dari tabel di atas. Relasi ini dapat dibagi ke dalam n fragment yang berbeda, di mana berisikan tuple-tuple dari rekening yang dimiliki oleh sebuah cabang utama. Jika bank hanya memiliki dua cabang, Hillside dan Valleyview maka ada dua fragment yang berbeda.

   Kemudian fragmentasi horizontal dapat diuraikan sbb :

   $Deposit_1 = \sigma_{\text{branch-name = "Hillside"}}(Deposit)$
   $Deposit_2 = \sigma_{\text{branch-name = "Valleyview"}}(Deposit)$

   Fragment $deposit_1$ disimpan pada site Hiilside dan fragment $deposit_2$ disimpan pada site Valleyview.

   Dua fragment ini digambarkan sbb:

branch-name	account-number	customer-name	balance
Hillside	305	Lowman	500
Hillside	226	Camp	336
Hillside	115	Khan	62

(a) $deposit_1$

branch-name	account-number	customer-name	balance
Valleyview	177	Camp	205
Valleyview	402	Khan	10000
Valleyview	408	Khan	1123
Valleyview	639	Green	750

(b) $deposit_2$

Fragmentasi di atas memenuhi kondisi lengkap jika "Hillside" dan "Valleyview" adalah harga-harga yang mungkin dari atribut branch-name. Kondisi penyusunan kembali

sangat mudah untuk diperiksa karena selalu mungkin untuk disusun kembali relasi global deposit melalui operasi berikut :

$$\text{Deposit} = \text{deposit}_1 \cup \text{deposit}_2$$

Begitu juga untuk kondisi disjoin. Kita akan memakai predikat yang digunakan dalam operasi seleksi yang mendefinisikan sebuah fragment kualifikasinya :

q_1 : branch-name = "Hillside"
q_2 : branch-name = "Valleyview"

2. *Fragmentasi Vertikal :*
   Pada fragmentasi vertikal, relasi deposit memerlukan penambahan tuple-id

   Berikut ini adalah relasi deposit dengan penambahan tuple-id :

branch-name	account-number	customer-name	balance	tuple-id
Hillside	305	Lowman	500	1
Hillside	226	Camp	336	2
Valleyview	177	Camp	205	3
Valleyview	402	Khan	10000	4
Hillside	115	Khan	62	5
Valleyview	408	Khan	1123	6
Valleyview	639	Green	750	7

   Sebuah fragmentasi vertikal dari relasi ini dapat diuraikan sebagai berikut :

   $\text{Deposit}_3 = \pi_{\text{branch-name,customer-name,tuple-id}}(\text{deposit})$
   $\text{Deposit}_4 = \pi_{\text{account-number,balance,tuple-id}}(\text{deposit})$

branch-name	customer-name	tuple-id
Hillside	Lowman	1
Hillside	Camp	2
Valleyview	Camp	3
Valleyview	Khan	4
Hillside	Khan	5
Valleyview	Khan	6
Valleyview	Green	7

(a) relasi deposit_3

account-number	Balance	tuple-id
305	500	1
226	336	2
177	205	3
402	10000	4
115	62	5
408	1123	6
639	750	7

(b) relasi $deposit_4$

Untuk menyusun kembali relasi deposit yang asli dari fragment-fragment, kita dapat menggunakan :

$$\pi \text{ Deposit-scheme}(deposit_3 \quad deposit_4)$$

Atribut join dari ekspresi di atas adalah tuple-id. Karena tuple-id menggambarkan sebuah alamat, hal ini memungkinkan untuk memasangkan sebuah tuple dari $deposit_3$ yang berhubungan dengan tuple dari $deposit_4$ dengan menggunakan alamat yang diberikan oleh harga tuple-id.

3. *Fragmentasi Campuran :*
Misalkan relasi r adalah relasi deposit dari gambar 1 di atas. Relasi ini dibagi ke dalam fragment $deposit_3$ dan $deposit_4$ seperti didefinisikan di atas. Selanjutnya kita dapat membagi fragment $deposit_3$ menjadi fragment $deposit_{3a}$ dan fragment $deposit_{3b}$ dengan menggunakan skema fragmentasi horizontal ke dalam dua fragment berikut :

$Deposit_{3a} = \sigma$ branch-name = "Hillside" $(Deposit_3)$
$Deposit_{3b} = \sigma$ branch-name = "Valleyview" $(Deposit_3)$

branch-name	customer-name	tuple-id
Hillside	Lowman	1
Hillside	Camp	2
Hillside	Khan	5

(a)Relasi $deposit_{3a}$

branch-name	customer-name	tuple-id
Valleyview	Camp	3
Valleyview	Khan	4
Valleyview	Khan	6
Valleyview	Green	7

(b) relasi $deposit_{3b}$

Replikasi :

Sistem memelihara beberapa salinan (copy) dari relasi. Setiap salinan disimpan pada beberapa lokasi yang berbeda

Replikasi & Fragmentasi

Rancangan ini merupakan kombinasi dari replikasi dan fragmentasi. Relasi dipartisikan ke dalam beberapa bagian. Sistem memelihara salinan yang identik untuk setiap bagian.