Tugas Basis Data Sesi 4

Aldi Maulana Iqbal – 2021080122

Data Models

Pengertian

Data model merupakan suatu metode konseptual untuk menata data dalam suatu sistem informasi. Data model menyediakan struktur untuk menyimpan data dan menentukan bagaimana data tersebut dapat dikelola dan diakses. Data model juga memungkinkan sekumpulan operator manipulatif untuk ditentukan pada data yang dimodelkan, sehingga memudahkan pengolahan data. Dan data model juga menerapkan serangkaian batasan untuk memastikan keakuratan data. Data model membantu untuk mengatur data secara konseptual sehingga memudahkan dalam pemahaman dan pengelolaan data.

Ada tiga jenis model data yang biasanya dikenal: model data tingkat tinggi (atau konseptual), model data tingkat rendah (atau fisik), dan model data representasional (atau model data basis rekaman).

1. Model data tingkat tinggi (atau konseptual) adalah model data yang menggambarkan struktur data secara konseptual, tanpa mempertimbangkan implementasi fisik database. Model data tingkat tinggi ini biasanya digunakan oleh pengembang sistem untuk memahami struktur data secara umum dan untuk menentukan bagaimana data akan disimpan dan diakses dalam sistem.
2. Model data tingkat rendah (atau fisik) adalah model data yang menggambarkan struktur data secara fisik, yaitu menggambarkan bagaimana data akan disimpan dalam media penyimpanan fisik seperti hard disk. Model data tingkat rendah ini biasanya digunakan oleh administrator database untuk mengelola database secara efektif.
3. Representasional atau model data basis rekaman adalah model data yang menggambarkan struktur data sebagai sekumpulan record (rekaman) dengan setiap record terdiri dari beberapa field (kolom). Model data ini biasanya digunakan oleh aplikasi-aplikasi yang mengakses database untuk mengolah data.

Hierarchical Model

Model hierarki adalah salah satu model basis data yang menggunakan struktur pohon untuk menyimpan data. Setiap catatan di dalam model hierarki terhubung satu sama lain melalui link, membentuk pohon yang terdiri dari simpul dan cabang.

1. Record: Kumpulan atribut yang masing-masing berisi satu nilai data.

2. Link: Asosiasi antara dua record.

Pohon berakar adalah pohon yang memiliki simpul akar yang merupakan simpul kosong atau dummy, yang menjadi dasar dari pohon tersebut. Kumpulan

pohon berakar dapat dikenal sebagai hutan. Model hierarki merupakan salah satu model basis data yang cukup populer karena mudah diimplementasikan dan memiliki waktu akses yang cepat untuk operasi pemilihan data tertentu. Namun, model ini memiliki beberapa kelemahan, seperti kesulitan dalam mengimplementasikan operasi pembaruan data dan keterbatasan dalam pemodelan beberapa jenis relasi data.

Tree Structure Diagrams

Tree structure diagrams adalah diagram yang menggambarkan struktur hierarkis dari suatu data atau informasi. Setiap item di diagram ini disebut node, dan node tersebut dapat memiliki child nodes (node anak). Child nodes dapat pula memiliki child nodes lainnya, membentuk struktur pohon yang terdiri dari node-node yang terkait secara hierarkis. Tree structure diagrams sering digunakan untuk menyimpan dan menampilkan data yang memiliki struktur hierarkis, seperti struktur direktori pada sistem file atau struktur organisasi perusahaan.

Struktur pohon terdiri dari dua komponen dasar yaitu rectangular boxes atau node dan line atau tautan. Rectangular boxes atau node merupakan bagian dari struktur pohon yang menyimpan data atau informasi. Setiap node pada struktur pohon dapat memiliki child nodes atau node anak yang terkait dengannya. Line atau tautan merupakan bagian dari struktur pohon yang menghubungkan node-node yang terkait satu sama lain. Line atau tautan ini menunjukkan relasi hierarkis antara node-node tersebut, sehingga membentuk struktur pohon yang terdiri dari node-node yang terkait secara hierarkis.

Diagram struktur pohon dapat digunakan untuk menentukan keseluruhan struktur logis dari basis data. Diagram struktur pohon menunjukkan bagaimana data atau informasi terorganisir dan terkait satu sama lain dalam suatu basis data. Diagram struktur pohon dapat digunakan sebagai diagram E-R (Entity-Relationship) dalam model hubungan entitas, yang menunjukkan hubungan antara entitas (atau objek) dalam basis data. Diagram struktur pohon juga dapat digunakan untuk menentukan struktur tabel dalam basis data relasional, yang menggambarkan bagaimana data diorganisir dalam tabel-tabel dan bagaimana tabel-tabel tersebut terkait satu sama lain melalui relasi.

Hubungan yang ada antara parent node dan child nodes pada struktur pohon harus berupa hubungan satu-ke-banyak (one-to-many) atau satu-ke-satu (one-to-one). Hubungan satu-ke-banyak (one-to-many) menunjukkan bahwa sebuah parent node dapat memiliki banyak child nodes, tetapi setiap child node hanya memiliki satu parent node. Contohnya, dalam struktur pohon yang menggambarkan struktur direktori pada sistem file, sebuah folder dapat memiliki banyak file di dalamnya, tetapi setiap file hanya terdapat di dalam satu folder.

Hubungan satu-ke-satu (one-to-one) menunjukkan bahwa sebuah parent node hanya memiliki satu child node, dan setiap child node hanya memiliki satu parent node. Contohnya, dalam struktur pohon yang menggambarkan struktur organisasi

perusahaan, seorang manager hanya memiliki satu assistant, dan setiap assistant hanya bekerja untuk satu manager.

Pada umumnya, struktur pohon yang menggunakan hubungan satu-ke-banyak lebih banyak digunakan daripada struktur pohon yang menggunakan hubungan satu-ke-satu. Hal ini karena struktur pohon yang menggunakan hubungan satu-ke-banyak lebih fleksibel dan dapat mengakomodasi data yang memiliki struktur hierarkis yang lebih kompleks.

Tautan atau hubungan antara parent node dan child nodes pada struktur pohon diwakili oleh garis dengan panah. Panah pada garis tersebut menunjukkan arah dari hubungan tersebut, yaitu dari parent node ke child nodes. Parent node mungkin memiliki panah yang menunjuk ke child nodes-nya, tetapi child nodes harus memiliki anak panah yang menunjuk ke parent node-nya.

Hal ini menunjukkan bahwa hubungan antara parent node dan child nodes pada struktur pohon selalu terjadi dalam satu arah, yaitu dari parent ke child. Parent node merupakan node yang terletak pada level yang lebih tinggi dalam struktur pohon, sedangkan child nodes merupakan node yang terletak pada level yang lebih rendah. Hubungan ini menggambarkan bahwa child nodes merupakan turunan dari parent node, dan terikat oleh hubungan hierarkis dengan parent node tersebut.

Operations dalam Hierarchical Data Model

Ada beberapa operasi dasar yang dapat dilakukan pada struktur data hirarkis, yaitu:

1. Insertion: Operasi ini merupakan proses penambahan data baru ke dalam struktur pohon. Data baru dapat ditambahkan pada posisi apa saja dalam struktur pohon, tergantung pada kondisi yang diinginkan.
2. Deletion: Operasi ini merupakan proses penghapusan data dari struktur pohon. Data yang dihapus bisa merupakan node yang tidak memiliki child nodes (node daun), atau node yang memiliki child nodes (node internal).
3. Updation: Operasi ini merupakan proses pembaruan data yang ada dalam struktur pohon. Data yang diperbarui bisa merupakan data yang terdapat pada node, atau data yang terkait dengan tautan atau hubungan antar node.
4. Retrieval: Operasi ini merupakan proses pengambilan atau pengambilan data dari struktur pohon. Data yang diambil bisa merupakan data yang terdapat pada node, atau data yang terkait dengan tautan atau hubungan antar node.

Semua operasi ini dapat dilakukan dengan menggunakan algoritma yang sesuai, tergantung pada kebutuhan dan kondisi yang diinginkan. Algoritma yang sering digunakan untuk melakukan operasi dasar pada struktur pohon antara lain adalah traversal, searching, dan sorting.

Query Language for Hierarchical Databases

Program Work Area

Program Work Area (PWA) adalah area memori yang digunakan oleh suatu program untuk menyimpan data sementara yang digunakan dalam proses eksekusi program. PWA biasanya digunakan untuk menyimpan data yang digunakan oleh suatu program untuk melakukan operasi-operasi tertentu, seperti data input, data hasil proses, atau data yang digunakan untuk mengeksekusi logika program.

Record template adalah template yang digunakan untuk mendefinisikan struktur data dari suatu record. Record template menyatakan jenis data dari setiap field dalam record, panjang field, dan urutan field dalam record.

Currency pointers adalah pointer yang menunjukkan posisi record saat ini dalam suatu set data. Currency pointers digunakan untuk menunjukkan record yang sedang aktif, yang digunakan dalam operasi-operasi seperti navigasi record, edit record, atau proses lainnya.

Status flag adalah flag atau bendera yang digunakan untuk menyimpan status dari suatu proses atau operasi. Status flag biasanya digunakan untuk menyimpan informasi seperti apakah proses berhasil atau tidak, atau apakah suatu kondisi terpenuhi atau tidak.

Get Command

Untuk mendapatkan data dari basis data hierarkis, digunakan perintah-perintah seperti `GET`, `GET FIRST`, dan `GET NEXT`.

1. Perintah `GET` digunakan untuk mengambil satu atau beberapa baris dari tabel berdasarkan kondisi tertentu.
2. Perintah `GET FIRST` digunakan untuk mengambil baris pertama dari tabel.
3. Perintah `GET NEXT` digunakan untuk mengambil baris berikutnya dari tabel setelah baris yang saat ini ditunjuk oleh pointer.

Perintah-perintah ini biasanya digunakan dalam kombinasi dengan perintah-perintah lain seperti `LOOP` atau `WHILE` untuk mengambil beberapa baris dari tabel.

Update Commands

Update command digunakan untuk mengubah data yang sudah ada di dalam database.

1. Insert command adalah perintah yang digunakan untuk menambahkan catatan baru ke dalam database.
2. Replace command adalah perintah yang digunakan untuk mengganti catatan yang sudah ada di dalam database dengan catatan baru. Cara penggunaannya mirip dengan insert command, namun replace command

akan menghapus catatan lama dan menambahkan catatan baru ke dalam database.
3. Delete command adalah perintah yang digunakan untuk menghapus catatan yang sudah ada di dalam database.

Virtual Records

Diagram struktur pohon tidak dapat mewakili hubungan many-to-many secara langsung. Oleh karena itu, untuk mewakili hubungan-hubungan tersebut dan tetap menjaga struktur pohon, harus menyalin data. Namun, replikasi data ini memiliki kelemahan seperti memerlukan lebih banyak penyimpanan dan memperbesar risiko terjadinya kesalahan atau kekeliruan pada data. Untuk mengatasi kelemahan tersebut, dapat menggunakan rekaman virtual.

Rekaman virtual adalah rekaman yang tidak memiliki nilai data, tetapi hanya berisi pointer ke rekaman fisik. Dengan menggunakan rekaman virtual ini, dapat menghindari replikasi dan hanya perlu menyimpan satu rekaman saja, serta menempatkan rekaman virtual alih-alih rekaman asli.

Rekaman virtual dapat membantu menjaga konsistensi data, tetapi masih terdapat masalah yang serius yaitu terjadinya pemborosan ruang penyimpanan. Rekaman virtual hanya berisi pointer ke rekaman fisik, sehingga tidak menyimpan nilai data secara langsung. Namun, rekaman fisik harus tetap disimpan, sehingga dapat terjadi pemborosan ruang penyimpanan jika terdapat banyak rekaman virtual yang merujuk ke satu rekaman fisik. Oleh karena itu, meskipun rekaman virtual dapat membantu menjaga konsistensi data, penggunaannya harus dikoordinasikan dengan baik untuk menghindari pemborosan ruang penyimpanan.

Implementation of Tree Structure Diagram

Untuk mengoptimalkan diagram struktur pohon, dapat digunakan leftmost-child, preorder threads dan next-sibling pointers sebagai gantinya dari parent-child pointers.

1. Leftmost-child pointer menunjukkan simpul anak pertama dari suatu simpul. Dengan menggunakan leftmost-child pointer, proses traversal dapat dilakukan dengan lebih efisien karena tidak perlu mengecek apakah simpul memiliki anak atau tidak.

2. Preorder thread pointer menunjukkan simpul selanjutnya dalam traversal preorder. Dengan menggunakan preorder thread pointer, proses traversal dapat dilakukan tanpa perlu mengecek apakah suatu simpul memiliki anak atau tidak.

3. Next-sibling pointer menunjukkan simpul saudara selanjutnya dari suatu simpul. Dengan menggunakan next-sibling pointer, proses traversal dapat dilakukan tanpa perlu mengecek apakah suatu simpul memiliki saudara atau tidak.

Mendefinisikan leftmost-child pointer, preorder thread pointer, dan next-sibling pointer akan membuat proses traversal lebih efisien dibandingkan dengan hanya menggunakan parent-child pointer. Hal ini karena kedua pointer tadi memungkinkan untuk traversal tanpa perlu mengecek apakah simpul memiliki anak, saudara, atau tidak.

Keuntungan Hierarchical Model

Berikut ini adalah keuntungan utama dari model data hierarkis:

1. Kemudahan: Dalam model data hierarkis, record terkait dalam bentuk hubungan parent/child. Struktur pohon ini memudahkan dalam menjalankan berbagai operasi seperti penambahan, penghapusan, dll. Hal ini menyebabkan desain basis data yang dihasilkan dari model ini menjadi sederhana dan mudah dipahami.

2. Integritas data: Dalam model data hierarkis, relasi parent/child antara berbagai record diwakili oleh relasi atau link. Setiap child segments hanya dapat terhubung ke satu parent, dan child hanya dapat diakses melalui parent. Hal ini membantu menjaga integritas data karena hanya ada satu cara untuk mengakses data child, yaitu melalui parent. Sehingga, jika terjadi perubahan pada data parent, akan mempengaruhi data child yang terhubung ke parent tersebut. Dengan demikian, model ini dapat membantu menjaga integritas data dengan mengontrol akses dan perubahan data.

3. Keamanan data: Dalam model data hierarkis, setiap segment anak hanya dapat terhubung ke satu parent, dan anak hanya dapat diakses melalui parent. Jadi, untuk menghapus segment anak, diperlukan informasi yang tepat dari segment parent. Hal ini memberikan tingkat keamanan data yang tinggi karena sistem manajemen basis data (DBMS) hanya akan mengizinkan akses dan perubahan data kepada pengguna yang memiliki otorisasi yang tepat. Dengan demikian, model ini dapat membantu menjaga keamanan data dengan membatasi akses ke data hanya pada pengguna yang berwenang.

4. Efisiensi: Model hierarkis mengandung hubungan satu ke banyak antara parent dan anak. Ketika basis data berisi banyak hubungan satu ke banyak antara berbagai rekaman, model ini akan menangani hal tersebut dengan sangat efisien. Karena keefisienan dalam penyimpanan data dan akses data dari relasi one-to-many yang diwakili oleh parent-child links. Dengan demikian, sistem dapat dengan cepat mengambil data dari basis data karena mereka terorganisir dalam pohon yang efisien dengan cara dalam mencari dan mengambil data.

5. Model data hierarkis sangat efisien dalam menangani jumlah besar transaksi. Hal ini terutama karena link atau hubungan yang dibuat oleh pointer pada berbagai rekaman bersifat permanen dan tidak dapat

dimodifikasi. Ini membuat sistem dapat dengan cepat mengambil data karena mereka terorganisir dalam pohon yang efisien dan dapat diakses dengan cepat. Selain itu, dengan tautan yang permanen, tidak perlu waktu yang banyak untuk membuat atau memodifikasi hubungan antara rekaman selama proses transaksi. Hal ini membuat proses transaksi lebih cepat dan efisien dibandingkan dengan model data lain.

Kelemahan Hierarchical Model

Berikut ini adalah kelemahan dari model data hierarkis:

1. Diperlukan pengetahuan tentang tingkat fisik penyimpanan data: Dalam model data hierarkis, hubungan satu ke banyak antara parent dan child dapat menyebabkan redundansi data, sehingga data harus disimpan di satu tempat dan direferensikan oleh tautan atau pointer fisik. Hal ini mengharuskan pengetahuan tentang tingkat fisik penyimpanan data, seperti pengetahuan tentang media penyimpanan, struktur file, dan alokasi ruang penyimpanan. Hal ini menyulitkan bagi pengelola yang tidak memiliki keahlian teknis. Karena itu sangat penting bagi pengelola basis data untuk memahami cara data disimpan secara fisik sehingga dapat mengoptimalkan penggunaan ruang penyimpanan dan memastikan performa yang baik.

2. Kompleksitas: Dalam model data hierarkis, tautan fisik atau pointer yang digunakan untuk menghubungkan berbagai rekaman dapat membuat database sulit untuk diperluas atau dimodifikasi. Karena setiap perubahan dalam struktur data, misalnya menambah atau menghapus field atau mengubah relasi antar rekaman, memerlukan modifikasi pada tautan fisik yang digunakan. Hal ini dapat menyulitkan pengelola basis data yang tidak memiliki keahlian teknis karena dapat memerlukan upaya penulisan ulang yang substansial dari kode program yang digunakan untuk mengakses dan memanipulasi data. Hal ini dapat membuat proses pengelolaan basis data lebih kompleks dan lebih sulit untuk diadministrasikan.

3. Ketidakfleksibelan: Salah satu kelemahan utama dari model data hierarkis adalah kurang fleksibel dalam menangani berbagai jenis hubungan yang terjadi dalam dunia nyata. Model ini didasarkan pada hubungan satu ke banyak antara rekaman yang diwakili oleh pointer yang permanen dan tidak dapat dimodifikasi. Namun dalam beberapa kasus, seperti hubungan banyak ke banyak, model ini kurang cocok dan sulit untuk digunakan. Model ini dapat menjadi kurang efektif jika menangani relasi yang lebih kompleks, atau jika perlu untuk menambahkan atau menghapus hubungan di kemudian hari. Hal ini dapat membuat model data hierarkis kurang cocok untuk aplikasi yang memerlukan keterbatasan atau perubahan dalam relasi data.

4. Kurangnya fasilitas query: Model data hierarkis memiliki keterbatasan dalam hal fasilitas query, ini dikarenakan kurangnya fasilitas query deklaratif (misalnya SQL) yang banyak digunakan dalam model data relasional. Sebaliknya, untuk mengakses informasi yang dibutuhkan, harus melakukan navigasi pointer yang rumit dan memerlukan pengetahuan tentang struktur data. Hal ini dapat membuat proses query menjadi lebih rumit dan membutuhkan waktu lebih lama untuk menemukan informasi yang dibutuhkan. Selain itu kurangnya fasilitas query adhoc yang mudah diakses dalam model data hierarkis, membuat model ini kurang cocok untuk aplikasi yang memerlukan kinerja query yang cepat dan fleksibel.

5. Masalah manajemen basis data: Dalam model data hierarkis, setiap perubahan dalam struktur data akan berdampak pada modifikasi yang signifikan pada program aplikasi yang mengakses basis data. Hal ini dapat membuat proses pengelolaan basis data lebih rumit dan menyulitkan. Selain itu, penambahan hubungan atau node baru juga akan menghasilkan tugas manajemen sistem yang lebih kompleks, seperti mengatur tautan pointer atau relasi antar node. Hal ini dapat membuat proses pengelolaan basis data lebih sulit dan lebih memakan waktu.

6. Masalah dengan operasi manipulasi data: Dalam model data hierarkis, melakukan operasi seperti penambahan, penghapusan, dan perubahan data dapat menyebabkan masalah yang rumit. Proses ini dapat mengubah struktur pointer dan tautan yang digunakan untuk menghubungkan rekaman, sehingga dapat menyebabkan kesalahan atau kekeliruan dalam data. Selain itu, proses pengambilan data juga cukup kompleks dan asimetris karena harus melakukan navigasi pointer untuk mengakses data yang diinginkan. Oleh karena itu, diperlukan model data yang lebih baik untuk mengatasi masalah ini, seperti model relasional yang menyediakan fasilitas query deklaratif dan menghindari redundansi data sehingga lebih mudah dikelola.

7. Kurangnya standar : Model data hierarkis tidak memiliki standar yang diterima secara luas atau spesifik untuk desain dan pemodelan basis data. Hal ini berarti bahwa implementasi dari model ini dapat berbeda dari satu DBMS ke DBMS lainnya. Hal ini dapat membuat proses pengembangan dan pengelolaan aplikasi menjadi lebih sulit dan memerlukan upaya khusus untuk menyesuaikan dengan setiap DBMS yang digunakan. Selain itu kurangnya standar dapat membuat lebih sulit bagi tim yang bekerja pada proyek yang sama untuk bekerja secara efektif, karena mereka harus memahami cara khusus DBMS bekerja dalam hal pemodelan data hierarkis.

Sistem Basis Data Hierarkis yang Tersedia Secara Komersial

Ada beberapa sistem basis data yang tersedia secara komersial yang menggunakan model data hierarkis. Beberapa contohnya adalah sebagai berikut:

1. IBM's Information Management System (IMS) adalah salah satu sistem basis data hierarkis yang dikembangkan oleh IBM. IMS menyediakan fasilitas manajemen data yang lengkap, termasuk pengelolaan data transaksi, navigasi data, dan pemrosesan data.
2. MRI's System 2000 adalah sistem basis data hierarkis yang dikembangkan oleh Magnetic Resonance Imaging. Sistem ini digunakan untuk mengelola data medis dan menyediakan fasilitas untuk pengambilan data, penyimpanan data, dan pemrosesan data.
3. IMS Informatics Mark IV adalah salah satu sistem basis data hierarkis yang dikembangkan oleh Informatics Corporation. Sistem ini digunakan untuk manajemen data transaksi dan menyediakan fasilitas untuk pengelolaan data yang efisien dan aman.
4. Time-shared Data Management System (TDMS) of SDC adalah sistem basis data hierarkis yang dikembangkan oleh SDC (System Development Corporation). Sistem ini digunakan untuk manajemen data dalam lingkungan waktu-berbagi dan menyediakan fasilitas untuk pengelolaan data, navigasi data, dan pemrosesan data.

Network Model

Model jaringan adalah salah satu model data yang menyediakan cara untuk menyimpan dan mengakses data yang terkait dalam suatu jaringan. Hal ini dilakukan dengan menggunakan sejumlah catatan yang dihubungkan satu sama lain dengan tautan atau hubungan. Tautan ini dapat menyatakan berbagai macam hubungan antara catatan, seperti satu ke banyak, banyak ke banyak, atau parent-child. Model ini sangat cocok digunakan untuk menyimpan data dengan struktur yang kompleks, seperti relasi antar entitas dalam aplikasi bisnis atau sistem manajemen inventori. Dengan model jaringan, data dapat diakses dengan cepat dan efisien dengan mengikuti tautan yang ada. Model ini merupakan pengembangan dari model hierarkis dan menawarkan fleksibilitas yang lebih baik dalam pemodelan data. Namun, model jaringan memiliki kekompleksan tinggi dibanding model hierarki dan mengharuskan pengetahuan yang baik dalam manajemen basis data untuk mengelola serta memodifikasinya.

Graph Structure Diagrams

Struktur grafik digunakan untuk merepresentasikan data dalam sebuah diagram dengan menggunakan kotak persegi panjang dan garis. Kotak persegi panjang mewakili catatan atau rekaman yang ada dalam database, sementara garis atau tautan mewakili hubungan antara dua catatan. Tautan ini tidak menyimpan nilai data, tapi hanya menunjukkan relasi antara dua catatan.

Diagram struktur grafik ini menentukan struktur logis dari database secara keseluruhan, yang menunjukkan bagaimana catatan terkait satu sama lain dalam basis data. Hal ini dapat digunakan untuk memodelkan relasi antar entitas dalam

aplikasi bisnis atau sistem manajemen inventori, dengan cara yang lebih fleksibel dan kompleks dari model hierarki. Namun, perlu pengetahuan yang baik tentang manajemen basis data dan sistem jaringan untuk dapat mengelola dan memodifikasi basis data yang dibangun dengan model grafik ini.

Operasi Dalam Network Data Model

Dalam model data jaringan, terdapat beberapa operasi dasar yang dapat dilakukan pada data, yaitu penambahan (insertion), penghapusan (deletion), pembaruan (updation), dan pengambilan data (retrieval)

1. Insertion : operasi penambahan digunakan untuk menambahkan catatan baru ke dalam jaringan, dengan menyatakan hubungan dengan catatan lain yang sudah ada.
2. Deletion : operasi penghapusan digunakan untuk menghapus catatan yang sudah tidak diperlukan lagi dari jaringan, dengan menghapus hubungannya dengan catatan lain.
3. Updation : operasi pembaruan digunakan untuk memperbarui data yang sudah ada dalam jaringan, baik dengan merubah data itu sendiri maupun dengan mengubah hubungannya dengan catatan lain.
4. Retrieval : operasi pengambilan data digunakan untuk mengambil data yang diperlukan dari jaringan, dengan mengikuti tautan yang tersedia.

Keuntungan Network Model

Network model memiliki keuntungan yang lebih baik dibandingkan dengan model hierarki dan menghilangkan beberapa batasan dari model hierarki. Beberapa keuntungan utama dari network model adalah:

1. Menghilangkan Redundansi Data: Model jaringan mampu menghilangkan redundansi data karena setiap catatan hanya perlu disimpan satu kali dalam basis data.
2. Perlu Penyimpanan Lebih Sedikit: Karena tidak ada redundansi data, maka model jaringan membutuhkan penyimpanan yang lebih sedikit dibandingkan dengan model hierarki.
3. Kinerja Lebih Baik: Model jaringan memberikan kinerja yang lebih baik karena data dapat diakses dengan lebih cepat dan efisien dengan mengikuti tautan yang ada.
4. Dapat Menangani Berbagai Jenis Hubungan: Model jaringan dapat menangani berbagai jenis hubungan seperti satu ke banyak, banyak ke banyak, dll.
5. Akses Data yang Mudah: Model jaringan memungkinkan akses data yang lebih mudah karena tidak perlu melakukan navigasi pointer seperti dalam model hierarki.

6. Mendorong Integritas Data: Model jaringan dapat mendorong integritas data karena menetapkan aturan yang ketat untuk menentukan hubungan antara cat

Sistem Basis Data Jaringan yang Tersedia Secara Komersial

Ada sejumlah sistem Database berdasarkan model jaringan. Beberapa diantaranya adalah sebagai berikut :

1. UNIVAC's DMS 1100 adalah sistem Database yang dikembangkan oleh UNIVAC (UNIVersal Automatic Computer) pada tahun 1960-an. DMS 1100 digunakan untuk mengelola data pada lingkungan mainframe dan digunakan oleh perusahaan besar dan pemerintah.

2. Cullinane's IDMS (Integrated Data Store) adalah sistem Database yang dikembangkan oleh Cullinane Database Systems pada tahun 1970-an. IDMS digunakan untuk mengelola data pada lingkungan mainframe dan digunakan oleh perusahaan besar dan pemerintah.

3. Cincom's TOTAL adalah sistem Database yang dikembangkan oleh Cincom Systems pada tahun 1970-an. TOTAL digunakan untuk mengelola data pada lingkungan mainframe dan digunakan oleh perusahaan besar dan pemerintah.

4. IBM's DBOMP (Database Oriented Mainframe Programming) adalah sistem Database yang dikembangkan oleh IBM pada tahun 1970-an. DBOMP digunakan untuk mengelola data pada lingkungan mainframe dan digunakan oleh perusahaan besar dan pemerintah.

5. Honeywell's Integrated Data Store (IDS) adalah sistem Database yang dikembangkan oleh Honeywell pada tahun 1970-an. IDS digunakan untuk mengelola data pada lingkungan mainframe dan digunakan oleh perusahaan besar dan pemerintah.

Semua sistem-sistem tersebut sudah tidak digunakan lagi dan digantikan oleh sistem Database yang lebih baru dan canggih seperti sistem Database relasional dan sistem Database NoSQL.

Relational Model

Struktur data dalam model relasional menggunakan konsep tabel yang terdiri dari baris dan kolom. Kolom mewakili atribut dari data, sementara baris mewakili entitas yang mewakili data. Setiap tabel memiliki kunci primer yang digunakan untuk mengidentifikasi baris unik dalam tabel. Ini memungkinkan untuk menghubungkan tabel satu dengan yang lain melalui relasi.

Integritas data dalam model relasional dijamin melalui konsep normalisasi. Normalisasi adalah proses yang digunakan untuk mengekstrak data yang sama dari tabel dan menyimpannya dalam tabel yang berbeda. Ini memastikan bahwa data tidak ada duplikasi dan mudah untuk diperbarui.

Manipulasi data dalam model relasional dilakukan melalui perintah SQL (Structured Query Language). SQL digunakan untuk mengambil, menambahkan, memperbarui, dan menghapus data dari tabel. Ini juga digunakan untuk mengambil data dari beberapa tabel dan menyatukannya menjadi satu.

Secara keseluruhan, model relasional adalah cara yang efektif untuk menyimpan dan mengakses data dalam database. Ini memungkinkan untuk membuat koneksi antara tabel dan menjamin integritas data melalui normalisasi. SQL digunakan untuk mengakses dan memanipulasi data dengan mudah.

Definisi Relasi

Secara sederhana, relasi adalah hubungan antara dua atau lebih himpunan. Dalam model relasional, relasi ditunjukkan sebagai tabel yang terdiri dari baris dan kolom. Setiap baris mewakili entitas yang memiliki hubungan dengan entitas lain dalam tabel.

Dalam konteks matematika, relasi dapat diartikan sebagai himpunan n-tupel terurut yang terdiri dari n elemen dari himpunan domain yang berbeda. Derajat relasi adalah jumlah himpunan domain yang digunakan dalam relasi.

Contohnya, dalam database siswa, relasi antara tabel "siswa" dan "kelas" adalah bahwa setiap siswa terdaftar dalam kelas tertentu. Domain dari relasi ini adalah himpunan "siswa" dan himpunan "kelas", dan derajat relasi adalah 2.

Secara umum, relasi membantu untuk menghubungkan dan mengkaitkan data dari berbagai tabel dalam database relasional. Ini memungkinkan untuk mengambil data dari beberapa tabel sekaligus dan menggabungkannya menjadi satu untuk analisis yang lebih baik.

Struktur Data Database Relasional

Model relasional menggunakan tabel sederhana sebagai struktur data utama untuk menyimpan dan mengakses data dalam database. Ini berbeda dari model jaringan dan pohon yang lebih rumit dan kompleks.

Dalam model relasional, tabel adalah kumpulan data yang disimpan dalam bentuk matriks dengan baris dan kolom. Baris mewakili entitas atau data individu, sementara kolom mewakili atribut atau karakteristik dari entitas tersebut.

Berikut ini adalah struktur data dasar yang digunakan dalam basis data relasional.

1. Relation adalah kumpulan data yang disimpan dalam tabel dalam model relasional. Setiap relasi mewakili entitas atau objek yang memiliki hubungan dengan entitas lain dalam tabel. Dalam E-R model, ini disebut sebagai "entity set".
2. Tuple adalah baris dalam tabel yang mewakili entitas unik dalam relasi. Setiap tuple menyimpan data tentang entitas yang sama. Dalam E-R model, ini disebut sebagai "entity".

3. Attributes adalah kolom dalam tabel yang mewakili atribut atau karakteristik dari entitas yang diwakili oleh tuple. Setiap atribut memiliki domain yang menentukan jenis data yang diizinkan. Dalam E-R model, ini sama dengan atribut.
4. Domain adalah himpunan data yang diizinkan untuk digunakan dalam atribut. Ini dapat mencakup tipe data seperti angka, teks, atau tanggal. Dalam E-R model, ini sama dengan domain.
5. Tuple variable adalah nilai yang dapat diasosiasikan dengan atribut dalam tuple. Ini dapat digunakan dalam operasi matematika atau logika. Dalam E-R model, ini disebut sebagai "value".
6. Degree adalah jumlah atribut dalam relasi. Ini menentukan derajat dari relasi.
7. Cardinality adalah jumlah tuple dalam relasi. Ini menentukan jumlah entitas yang diwakili oleh relasi.

Jenis Atribut: Jenis atribut yang dibahas dalam model ER juga berlaku di sini.

1. Keys adalah atribut yang digunakan untuk mengidentifikasi tuple unik dalam relasi. Ada dua jenis kunci yaitu kunci primer dan kunci asing. Kunci primer digunakan untuk mengidentifikasi tuple unik dalam tabel, sementara kunci asing digunakan untuk menghubungkan tabel satu dengan yang lain melalui relasi.
2. Database instance adalah istilah yang digunakan untuk menggambarkan data yang disimpan dalam database pada saat ini. Ini mencakup semua data yang ada dalam tabel-tabel relasi.
3. Relation schema adalah struktur tabel yang digunakan untuk menyimpan data dalam relasi. Ini mencakup nama tabel, nama atribut, tipe data, dan kunci primer.
4. Relation instance adalah istilah yang digunakan untuk menggambarkan data yang disimpan dalam tabel pada saat tertentu. Ini mencakup semua data yang ada dalam tabel, termasuk tuple dan nilai atribut.

Integrity Constraints

Integrity Constraints atau batasan integritas adalah aturan atau pengaturan yang diterapkan pada database untuk memastikan bahwa data yang disimpan dalam database tetap stabil, akurat, dan konsisten. Ini dapat berupa aturan seperti kunci primer yang digunakan untuk mengidentifikasi tuple unik dalam tabel, atau batasan referensial yang digunakan untuk menghindari data yang hilang atau tidak konsisten saat menghubungkan tabel satu dengan yang lain. Ini memastikan bahwa data yang disimpan dalam database tetap benar dan dapat diandalkan.

1. Entity Integrity Rule (Integrity Rule 1) adalah aturan yang menyatakan bahwa kunci primer atau bagian darinya dalam relasi apa pun tidak boleh kosong atau nol. Ini berarti bahwa kunci primer harus memiliki nilai yang unik dan tidak boleh kosong. Hal ini dilakukan untuk menjamin bahwa setiap tuple dalam relasi dapat diidentifikasi dengan benar dan tidak ada duplikat.

2. Referential Integrity Rule (Integrity Rule 2) adalah aturan yang digunakan untuk menjamin integritas data saat menghubungkan tabel satu dengan yang lain melalui kunci asing. Ini memastikan bahwa setiap nilai kunci asing harus memiliki nilai yang sesuai dengan kunci utama yang terkait dengannya. Misalnya, jika atribut A adalah kunci utama dalam relasi R1 dan juga kunci asing dalam relasi R2, maka nilai A dalam relasi R2 harus sama dengan nilai A dalam relasi R1 atau null. Ini memastikan bahwa tidak ada data yang hilang atau tidak konsisten saat menghubungkan tabel satu dengan yang lain.
3. Domain Constraints adalah pembatasan yang diterapkan pada setiap nilai atribut dalam database. Pembatasan ini diterapkan pada domain atribut, yang menentukan jenis data yang diizinkan dalam atribut. Pembatasan ini termasuk tipe data, ukuran variabel, dan pemeriksaan yang digunakan untuk memastikan bahwa data yang dimasukkan dalam atribut sesuai dengan kriteria yang ditentukan. Ini bertujuan untuk menjaga konsistensi dalam database dan memastikan bahwa data yang disimpan dalam database valid dan dapat diandalkan.
4. Key Constraints adalah aturan yang diterapkan pada relasi untuk memastikan bahwa atribut primary key memiliki nilai unik. Atribut primary key digunakan untuk mengidentifikasi tuple unik dalam tabel. Oleh karena itu, tidak diperbolehkan adanya nilai duplikat dalam atribut primary key. Hal ini membuat data dalam tabel tetap konsisten dan dapat diandalkan. Ini juga membantu dalam proses pembuatan relasi antar tabel.
5. Tuple Uniqueness Constraints adalah batasan yang digunakan untuk memastikan bahwa tidak ada tuple duplikat dalam relasi. Ini berarti bahwa setiap tuple dalam relasi harus memiliki nilai yang unik dan tidak boleh ada dua tuple yang sama dalam relasi. Ini memastikan bahwa data yang disimpan dalam relasi valid dan konsisten, dan memudahkan proses analisis dan manipulasi data. Tuple Uniqueness Constraints dapat diterapkan dengan menggunakan kunci primer yang digunakan untuk mengidentifikasi tuple unik dalam relasi.

CODD's Rules

Dr. Edgar F. CODD menyarankan seperangkat aturan yang diperlukan untuk sistem untuk memenuhi syarat sebagai Sistem Manajemen Database Relasional. Aturan CODD, atau "Codd's Rules", terdiri dari 12 aturan yang digunakan untuk menentukan apakah suatu sistem dapat dianggap sebagai sistem manajemen database relasional yang baik. Aturan CODD adalah sebagai berikut:

1. Aturan Data Atomicity: Data harus dapat dibagi menjadi unit-unit yang tidak dapat dibagi lagi.
2. Aturan Data Consistency: Data harus konsisten dan sesuai dengan aturan yang ditentukan.

3. Aturan Data Isolation: Data harus dapat diisolasi dari pengaruh transaksi lain.
4. Aturan Data Durability: Data harus dapat bertahan dan tetap tersimpan setelah transaksi selesai.
5. Aturan Non-Subversion: Sistem harus dapat diandalkan dan tidak dapat digunakan untuk melakukan aksi yang tidak sah.
6. Aturan Access Control: Sistem harus dapat mengontrol akses ke data.
7. Aturan Comprehensive Data Sublanguage: Sistem harus menyediakan bahasa yang dapat digunakan untuk mengakses data secara komprehensif.
8. Aturan View Updating: Sistem harus dapat mengupdate data yang ditampilkan dalam pandangan.
9. Aturan High-level Insert, Update and Delete: Sistem harus menyediakan perintah yang dapat digunakan untuk menambah, memperbarui, dan menghapus data dengan mudah.
10. Aturan Logical Data Independence: Sistem harus dapat mengubah struktur data tanpa mempengaruhi aplikasi yang mengakses data.
11. Aturan Physical Data Independence: Sistem harus dapat mengubah cara data disimpan tanpa mempengaruhi aplikasi yang mengakses data.
12. Aturan Distribution Independence: Sistem harus dapat digunakan pada berbagai sistem komputer yang terdistribusi tanpa perubahan pada aplikasi.

Operasi pada Model Relasional

Operasi dasar pada model data relasional adalah insertion, deletion, updation, dan retrieval.

1. **Insertion** digunakan untuk menambahkan record baru ke dalam tabel, tetapi terdapat beberapa kondisi yang harus diperhatikan seperti duplikat pada primary key, nilai NULL, nilai data pada atribut kunci asing yang tidak ada dalam atribut kunci primer, dan domain yang tidak sesuai.
2. **Deletion** digunakan untuk menghapus record dari tabel.
3. **Updation** digunakan untuk memperbarui nilai data dari suatu record dalam tabel, tetapi harus memperhatikan kondisi seperti nilai kunci asing yang sesuai dan nilai baru yang belum ada di tabel.
4. **Retrieval** digunakan untuk mencari dan mengambil record dari tabel.

Keunggulan Relational Model

Model Relasional memiliki beberapa keuntungan utama, di antaranya:

1. Simplicity: Model basis data relasional sangat mudah dan sederhana untuk dirancang dan diimplementasikan pada tingkat logika. Tabel-tabel yang berbeda dalam basis data dapat dengan mudah dirancang menggunakan atribut dan nilai data yang sesuai. Semua hubungan dirancang dalam bentuk tabel, yang membantu pengguna untuk fokus pada tampilan logika basis data daripada detail internal yang rumit tentang bagaimana data disimpan.
2. Flexible: Basis data relasional memberikan fleksibilitas yang memungkinkan perubahan dalam struktur basis data dengan mudah diterima.
3. Data Independence: Karena data berada dalam tabel, struktur basis data dapat diubah tanpa harus mengubah aplikasi yang didasarkan pada struktur tersebut. Jika menggunakan basis data non relasional, mungkin harus memodifikasi aplikasi yang akan mengakses informasi ini dengan menyertakan pointer ke data baru. Namun dengan basis data relasional, informasi segera dapat diakses karena secara otomatis terkait dengan data lain karena posisinya dalam tabel.
4. Structural Independence: Basis data relasional hanya mengenai data dan bukan struktur, yang meningkatkan performa. Oleh karena itu waktu pemrosesan dan ruang penyimpanan lebih besar dalam basis data relasional tetapi pengguna tidak perlu tahu detail desain struktur. Fleksibilitas struktur basis data relasional memungkinkan kombinasi data untuk diambil yang tidak pernah diantisipasi pada saat basis data awal dibuat.
5. Query Capability: Ini memungkinkan bahasa query tingkat tinggi yaitu SQL (Structure Query Language) yang menghindari navigasi basis data yang rumit. Dalam model ini, query didasarkan pada hubungan logika dan pemrosesan query tersebut tidak memerlukan jalur akses yang ditentukan sebelumnya di antara data yaitu pointer.
6. Matured Technology: Model relasional berguna untuk mewakili sebagian besar objek dunia nyata dan hubungan antar objek. Implementasi hubungan sangat mudah melalui penggunaan kunci.
7. Kemampuan untuk dengan mudah memanfaatkan teknologi perangkat keras baru yang membuat hal-hal lebih mudah bagi pengguna.

Kelemahan Relational Model

Kekurangan utama dari model basis data relasional adalah sebagai berikut:

1. Kinerja yang terbatas,
2. Keterbatasan dalam menangani objek besar biner,
3. Biaya overhead perangkat keras yang tinggi,
4. Kesulitan dalam memetakan objek ke database relasional,
5. Integritas data yang sulit dipastikan.

Sistem Basis Data Relasional yang Tersedia Secara Komersial

Beberapa contoh sistem basis data relasional yang tersedia secara komersial adalah Oracle, Sybase, MAGNUM, IBM's Query by example, dan NOMAD systems of NCSS.

Oracle adalah salah satu sistem basis data relasional yang paling populer dan banyak digunakan di dunia bisnis. Ini menyediakan berbagai fitur dan tools untuk membuat, mengelola, dan mengekstrak data dari basis data relasional.

Sybase adalah sistem basis data relasional yang dikembangkan oleh Sybase, Inc. Ini menyediakan berbagai fitur yang membantu dalam pengelolaan data, seperti pemrosesan transaksi, pembuatan laporan, dan analisis data.

MAGNUM adalah sistem basis data relasional yang dikembangkan oleh Magnasoft Corporation. Ini memiliki fitur yang memungkinkan untuk pengelolaan data, pembuatan laporan, dan analisis data.

IBM's Query by Example adalah sistem basis data relasional yang dikembangkan oleh IBM. Ini menyediakan fitur yang memungkinkan untuk pengelolaan data, pembuatan laporan, dan analisis data.

NOMAD systems of NCSS adalah sistem basis data relasional yang dikembangkan oleh NCSS. Ini menyediakan fitur yang memungkinkan untuk pengelolaan data, pembuatan laporan, dan analisis data.

Perbandingan DBMS dan RDBMS

DBMS	RDBMS
DBMS (Database Management System) adalah sistem software yang digunakan untuk mengelola basis data.	RDBMS (Relational Database Management System) adalah jenis DBMS yang mengelola basis data dengan menggunakan model relasional.
DBMS dapat digunakan untuk mengelola berbagai jenis basis data seperti basis data hierarki, basis data jaringan, dan basis data objek.	RDBMS hanya dapat digunakan untuk mengelola basis data relasional.
DBMS tidak memerlukan pengelompokan data dalam tabel-tabel yang saling terkait.	RDBMS memerlukan pengelompokan data dalam tabel-tabel yang saling terkait.
DBMS tidak selalu memerlukan penggunaan kunci primer dan kunci asing untuk mengatur relasi antar tabel.	RDBMS memerlukan penggunaan kunci primer dan kunci asing untuk mengatur relasi antar tabel.
DBMS dapat digunakan dalam berbagai jenis aplikasi seperti aplikasi pribadi, aplikasi bisnis, dan aplikasi sains.	RDBMS lebih cocok digunakan dalam aplikasi bisnis yang memerlukan pengelolaan data yang cepat dan efisien.

Perbandingan Model Data

Model Data	Deskripsi	Kelebihan	Kekurangan
Hierarchical	Data dalam model ini disusun dalam bentuk pohon, dengan satu node di atas sebagai parent dari node-node di bawahnya.	Struktur data yang jelas dan mudah dipahami.	Keterbatasan dalam menangani relasi antar data yang kompleks.
Network	Data dalam model ini disusun dalam bentuk jaring, dengan satu node dapat memiliki lebih dari satu parent dan child.	Fleksibilitas dalam menangani relasi antar data yang kompleks.	Struktur data yang lebih rumit dan kompleks dibandingkan dengan model hierarchical.
Relational	Data dalam model ini disusun dalam tabel-tabel yang saling terkait melalui kunci utama dan kunci asing.	Fleksibilitas dalam menangani relasi antar data yang kompleks.	Kinerja yang terbatas pada data besar dan kompleks.
Object-oriented	Data dalam model ini disusun dalam bentuk objek-objek yang saling terkait melalui relasi inheritance dan association.	Kapabilitas untuk menangani data yang kompleks dan bervariasi.	Implementasi yang lebih sulit dibandingkan dengan model relational.