BAB 3

SISTEM MANAJEMEN DATA (*DATA MANAGEMENT SYSTEMS*)

Tujuan Pembelajaran:

1. Memahami dasar alasan penggunaan pendekatan basis data
2. memahami komponen dasar dari konsep basis data
3. Mengetahui karakteristik tiga model basis data: hirarki, jaringan, dan relasional
4. Mengetahui fitur-fitur operasional terkait dengan risiko penyebaran model basis data terpusat, terpartisi, dan tereplikasi dalam lingkungan DDP
5. Mengetahui tujuan dan prosedur audit yang digunakan untuk menguji pengendalian manajemen basis data

1. Data Management Approaches

Manajemen data dapat dibagi ke dalam dua pendekatan secara umum: yaitu *flat-file model* dan *database model.*

A. The Flat-File Approach

The flat-file model sering dihubungkan dengan *legacy system* (sistem warisan). Sistem ini merupakan sistem *mainframe* yang besar, muncul pada akhir 1960-an sampai 1980-an. Flat-file model menggambarkan suatu suatu lingkungan yang *file* data individualnya tidak terkait dengan file lainnya. Dalam model ini *end user* tidak berbagi data dengan *end user* lainnya. Ketika *end user* lain membutuhkan data yang sama untuk tujuan yang berbeda, mereka harus menyusun set data yang terpisah untuk memenuhi kebutuhan mereka. Kelemahan model ini munculnya *data redudancy* yang mengakibatkan masalah pada *data storage, data updating*, dan informasi sekarang (*currency of information*).

1) Data Storage
   Suatu sistem informasi yang efisien akan menangkap dan menyimpan data hanya seketika dan membuat sumber tunggal tersedia untuk semua pengguna siapapun yang membutuhkan. Dalam lingkungan *Flat-file*, hal ini tidak mungkin. Untuk menemukan kebutuhan data pribadi para pemakai, organisasi harus membuat biaya-biaya kedua-duanya berbagai koleksi dan berbagai penyimpanan memeriksa prosedur.
2) Data Updating
   Organisasi menyimpan banyak data pada file acuan dan file master yang memerlukan pembaharuan berkala untuk mencerminkan perubahan. sebagai contoh, suatu perubahan suatu alamat atau nama pelanggan harus dicerminkan dalam master file yang tepat. Kapan para pemakai menyimpan file terpisah, semua perubahan harus dibuat secara terpisah untuk masing-masing pemakai. Ha ini secara signifikan menambahkan beban tugas dan kos manajemen data.
3) Currency of Information
   Kegagalan untuk membaharui semua file pemakai yang terpengaruh oleh perubahan dalam status. Jika pembaharuan informasi tidak disebarkan dengan baik, perubahan tidak akan tercermin dalam beberapa data

pemakai, sehingga menghasilkan keputusan berdasar pada informasi yang ketinggalan jaman.

4) Task Data Dependency (Limited Access)
masalah lain dengan pendekatan flat-file adalah ketidakmampuan pemakai untuk memperoleh informasi tambahan, ketika kebutuhannya berubah, yaitu ketegantungan task-data. Set informasi pemakai dibatasi oleh data yang dikuasai dan dikendalikannya. Oleh karena pemakai tidak saling berhubungan, maka sangat sukar untuk menetapkan suatu mekanisme untuk yang formal untuk pembagian data. Hal ini menyebabkan kebutuhan informasi baru memerlukan banyak waktu, menghalangi capaian, menambah pemborosan data, dan memicu kos manajemen data lebih tinggi. Selain itu, keterbatsan akses menghalangi pembagian data yang efektif di antara para pemakai.

5) Flat Files Data Integration (Limited Inclusion)
pendekatan Flat-file adalah single-view model. File disusun, diformat, dan diatur sesuai dengan kebutuhan pemilik atau pemakai utama data. Beberapa penyusunan ini kadangkala mengeluarkan/meniadakan atribut data yang berguna bagi pemakai lain, sehingga mencegah pengintegrasian data di antara organisasi. Pilihan yang dihadapi pemakai dalam masalah ini:
   1. Tidak menggunakan data akuntansi untuk mendukung keputusan
   2. Memanipulasi struktur data yang ada untuk disesuaikan dengan kebutuhan
      unik pemakai atau
   3. Memperoleh tambahan pribadi data dan membuat biaya-biaya dan permasalahan operasional berhubungan dengan pengulangan data.

   GAMBAR 3-1 FLAT-FILE MODEL (HLM 94)

B. The Database Approach

Untuk mengatasi permasalahan pada flat-file model digunakan pendekatan database untuk manajemen data.

Database management system (DBMS) adalah sistem *software* khusus yang diprogram untuk mengetahui apakah elemen data masing-masing pemakai diotorisasi untuk diakses. Program user mengirim permintaan data ke DBMS yang selanjutnya akan divalidasi dan diotorisasi akses ke *database* sesuai dengan tingkat otoritas pemakai. Jika pemakai meminta (mengakses) data yang bukan otorisasinya, maka permintaan ditolak. Pendekatan ini memusatkan pengorganisasian data ke dalam *database* umum sehingga dapat dibagi dengan pemakai lainnya. Hal ini mengatasi kelemahan pendekatan *flat-file model.*

1) Elimination of Data Storage Problem
Setiap elemen data hanya disimpan sekali, sehingga membatasi pengulangan data serta mengurangi pengumpulan data dan kos penyimpanan.

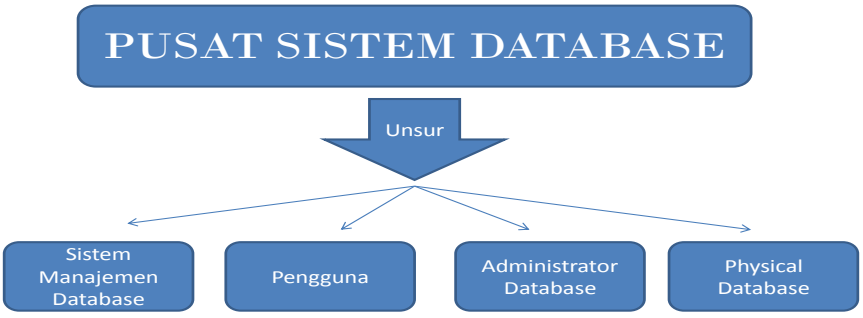
2) Elimination Data Update Problem
Oleh karena elemen data yang ada pada satu tempat maka untuk memperbaharui hanya memerlukan satu prosedur tunggal.

3) Elimination of Currency Problem
Perubahan tunggal pada atribut database yang dibuat secara otomatis tersedia untuk seluruh atribut user.

4) **Elimination of Task Data Dependency Problem**
Dengan akses penuh ke seluruh domain data entitas, perubahan pada kebutuhan informasi pemakai dapat memuaskan tanpa harus membuat set data khusus tambahan. Konstrain pemakai hanya pada keterbatasan ketersediaan data pada entitas dan legitimasi kebutuhan pemakai untuk mengakses.
5) **Elimination of Data Integration Problem**
Oleh karena data berada pada lokasi yang dapat diakses secara global dan umum, maka data diintegrasikan ke dalam seluruh aplikasi para pemakai.
GAMBAR 3-2 DBMS MODEL (HLM 96)

2. Centralized Database Systems

A. Database Management System



1) **Typical Features (Ciri Khusus)**
   a. **Program Development,** berisi Aplikasi Pengembangan Program Perangkat Lunak
   b. **Backup and Recovery,** sejak memproses, secara periodik DBMS membuat *backup copy* di *physical database*, dan jika terjadi bencana/kerusakan akan dapat memunculkan kembali
   c. Database Usage Reporting, ciri ini menyatakan data statistik apa yang digunakan, kapan mereka gunakan dan siapa yang menggunakan
   d. Database Access, ciri yang sangat penting yaitu memberikan ijin otorisasi untuk dapat mengakses, baik formal maupun informal *database*
   GAMBAR 3-3 ELEMEN OF DATABASE CONCEPT (HLM 98)

2) **Data Definition Language (DDL)**
DDL adalah program bahasa pengkodean yang digunakan untuk menetapkan database dalam DBMS. DDL mengidentifikasikan nama dan hubungan semua unsur data, dokumen/catatan, dan *file* yang terdapat di dalam *database*
GAMBAR 3-4 OVERVIEW OF DBMS OPERATION (HLM 99)

3) Database View (Schemas)

a. Internal View/Physical View

Menggambarkan struktur catatan data, hubungan-hubungan antara sebuah *file* dan rencana *physical* dan rangkaian catatan dalam *file*

b. Conceptual View/Logical View

Pandangan ini menggambarkan seluruh *database*. Ini terlihat mewakili logika *database* dan mengiktisarkan, daripada hal penyimpanan *physical*. Ini hanya suatu konsep pandangan untuk sebuah *database*.

c. External View/User View

Pandangan ini diartikan sebagai bagian dari pengguna-pengguna *database*. Bagian pengguna individual yang diotorisasi untuk mengakses untuk menngunakan dan menunjukkan *database*.

4) Formal Access: Application Interfaces

Ada dua jalan untuk pengguna mengakses *database*, salah satan unya yang mungkin adalah dengan aplikasi *formal interface*

a. Data Manipulation Language

adalah bahasa pemilik pemrograman yang khusus digunakan DBMS untuk mendapatkan kembali, memproses dan menyimpan data.

b. DBMS Operation

1) Pengguna program mengirimkan permintaan data untuk DBMS. Permintaan ditulis di sebuah bahasa manipulasi data khusus. Itu melekat di dalam pengguna program
2) Analisis DBMS mencocokan unsur-unsur data yang berlawanan dari pandangan pengguna dan pandangan konseptual. Jika data yang diminta sama, maka akan diproses, jika tidak sama akan ditiadakan.
3) DBMS menetapkan parameter struktur data dari pandangan internal dan itu tidak ketnggalan jaman dari sistem operasi ini, yang mana kinerja data aktual mendapatkan kembali informasinya
4) Penggunaan yang tepat metode akses, operasi sistem berinteraksi dengan alat penyimpanan untuk mendapatkan kembali data dari *database physical*
5) Sistem operasi ketika data disimpan dalam daerah penyangga di main memori diatur oleh DBMS
6) DBMS mentranfer data untuk lokasi pekerjaan pengguna-pengguna di main memori
7) Ketika pemrosesan lengkap, langkah nomer 4, 5 , 6 dikembalikan untuk memperbaiki data yang telah diproses ke dalam database.

5) Informal Access: Query Language

Metode akses kedua dari database adalah metode informal Query.

Query adalah metodologi akses khusus yang menggunakan perintah-perintah bahasa Inggris untuk membangun daftar atau informasi dasar yang lain dari sebuah *database*.

Pengguna dapat mengakses data melalui *Query* langsung, tanpa memerlukan program pengguna formal.

Query menyediakan fasilitas yang ramah lingkungan untuk mengintegrasikan dan mendapatkan kembali data untuk prosedur laporan khusus

a. SQL

SQL adalah generasi keempat, yang bukan bahasa prosedural (perintah bahasa Inggris), dengan banyak perintah yang memungkinkan pengguna untuk memasukkan, menemukan/mengambil data, dan memodifikasi data dengan mudah. Pilihan perintah merupakan sebuah peralatan yang kuat untuk menemukan data. SQL adalah peralatan proses data yang sangat efisien

b. QBE

Queries moderen lainnya adalah query by example (QBE). Dalam sistem GUI, pengguna dapat menggunakan dengan sederhana “drag dan drop” obyek untuk membangun query yang akan memperlihatkan hasilnya saat membuat pola. Metode ini membuat semua itu lebih mudah untuk pengguna akhir, untuk perkembangan bagian pekerjaan *query* dari pada dengan pengetahuan SQL .

GAMBAR 3-5 CONTOH ‘SELECT’ COMMAND USED TO QUERY AN INVENTORY DATABASE

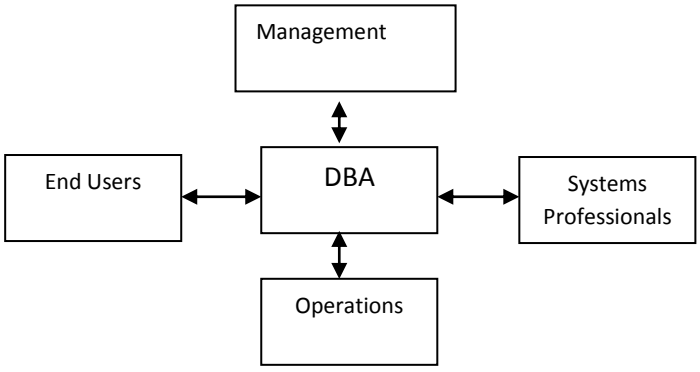
B. The Database Administrator (DBA)

1) *Database Administrator*, adalah tanggung jawab untuk mengelola sumber *database*. Posisi ini tidak ditemukan pada organisasi *flat*. Tugas *database administrator* al: membagi *database* umum yang diperlukan oleh berbagai *user* yg memerlukan pengorganisasian, koordinasi, aturan-aturan, dan pedoman untuk melindungi integritas *database*. Tugas DBA antara lain meliputi:

a. *Database planning*
b. *Database design*
c. *Database implementation*
d. *Database operation*
e. *Database maintenance*
f. *Database growth and change*

2) *Data Dictionary*. Salah satu tugas DBA adalah menciptakan dan menjaga *data dictionary*. *Data dictionary* mendeskripsikan setiap elemen data dalam *database. Data dictionary* memudahkan seluruh *user* berbagi pandangan sama atas sumber data (baik dalam bentuk kertas maupun *online*).

3) *Organizational Interaction of DBA*



Mengacu pada gambar 3-3 (hlm 98) dan gambar di atas, sejalan dengan kebutuhan informasi, *user* akan meminta secara formal aplikasi komputer kepada *system professional*. Permintaan tersebut ditangani melalui prosedur-prosedur pengembangan sistem formal. Dalam hal ini DBA berfungsi mengevaluasi permintaan untuk menentukan kebutuhan *database user*.

C. The Physical Database

1) *Physical Database*, adalah level paling rendah dari database dan satu-satunya level yang ada dalam bentuk *form. Physical database* terdiri dari titik-titik *magnetic* pada *magnetic disk*. Pada level fisik, database membentuk kumpulan logis catatan dan file yang terdapat dalam sumber data perusahaan.

- *Data Structure,* adalah 'bata' *database*, yang membolehkan catatn ditempatkan, disimpan, dipanggil, dan dimungkinkan dipindah dari catatan satu ke lainnya. Struktur data terdiri dari:
  a. Organisasi Data adalah cara pencatatan secara fisik yang diatur pada alat penyimpan *secondary*.
  b. *Data Access Method* adalah teknik yang digunakan untuk menempatkan catatan dan mengendalikan melalui *database*.

Tidak ada satupun struktur terbaik untuk tuas pemrosesan. Masing-masing ada *trade-offnya*. Beberapa hal berikut ini dapat menjadi pertimbangannya:

- ✓ Cepatnya akses file dan pemanggilan data
- ✓ Efisiensi penggunaan penyimpanan disk
- ✓ *High throughput* utk pemrosesan transaksi
- ✓ Melindungi data hilang
- ✓ Memudahkan pemulihan kegagalan sistem
- ✓ Pengakomodasian pertumbuhan file

2) *Data Hierarchy*, beberapa istilah penting dalam *data hierarchy*:

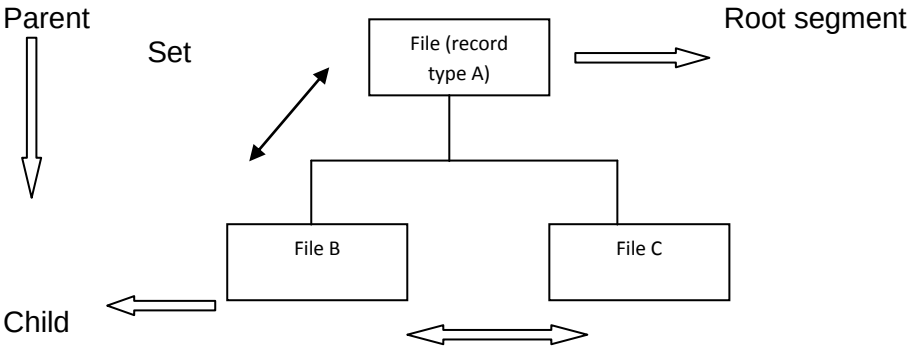
- *Data Attribute/Field* →item tunggal data misal: nama konsumen, akun rekening, alamat, dll.
- *Record*→kumpulan *field* yang mendeskripsikan karakteristik relevan kejadian entitas yang akan ditelusuri .
- *File/Entity*→sumber, kejadian, atau agen tunggal yang dipilih untuk mengumpulkan data misal: sediaan, aktivitas penjualan, *customer*, dll.
- *Database*→seperangkat tabel atau *file* yang bersama-sama membuat suatu aplikasi mampu melayani kebutuhan *user*nya terkait dengan proses atau fungsi bisnis tertentu.
- *Enterprise Database*→seperangkat umum file data atau tabel utk seluruh organisasi atau perusahaan, misal: *Oracle, Microsoft*, dll

D. Three DBMS Model

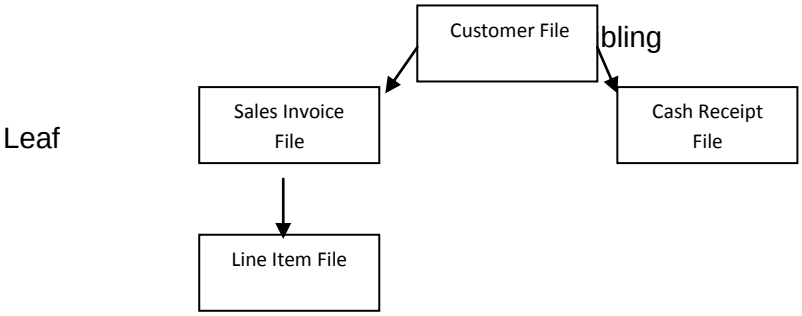
Data model adalah suatu representasi abstrak data mengenai entitas, termasuk sumber (aset), kejadian (transaksi) dan agen (personalia atau *customer*) dan hubungan mereka dalam organisasi.

Tujuan *Data Model* adalah menyajikan atribut entitas dengan cara yang mudah dipahami oleh pemakai. Ada tiga model data

1) *The Hierarchical Model* (=struktur pohon, gambar 3-8, hlm 107)



Contoh hirarki dalam DBMS:



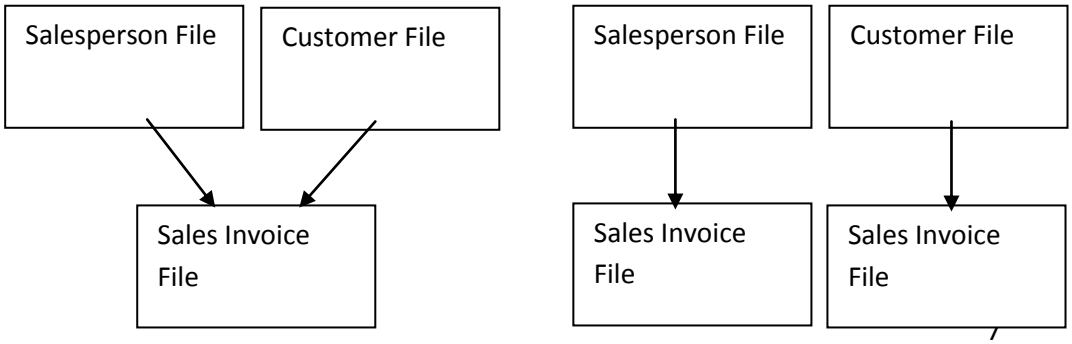
Model ini sering disebut *navigational database* karena membuatan garis file diikuti pra penetapan jalur (path). Hal ini ditetapkan melalui hubungan eksplisit antara catatan terkait. Cara mengakses data paling bawah dalam struktur pohon adalah dari akar dan melalui jalur navigasional pointer turun ke catatan yang diinginkan.
Model ini memiliki konstrain:

1. *Parent* dapat memiliki satu atau lebih catatan *child*.
2. Tidak ada satupun catatan *child* memiliki *parent* lebih dari satu.

Keterbatasan model ini:

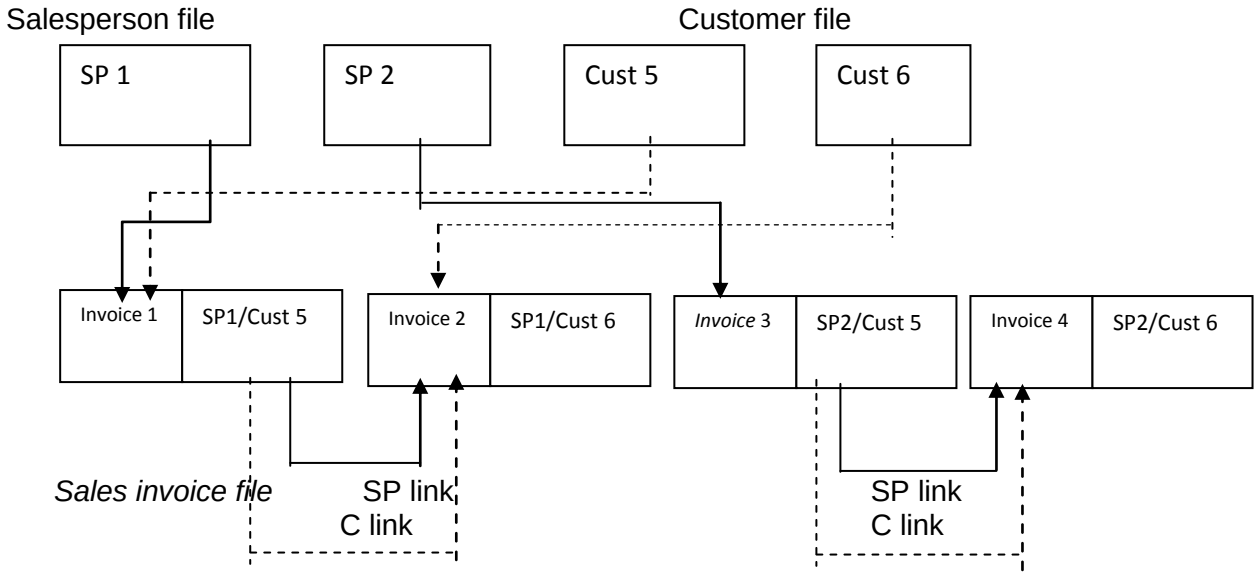
- ✓ Tidak mencerminkan kondisi sebenarnya
- ✓ Satu catatan *child* dapat memiliki *parent* lebih dari satu
  Untuk mengatasi kelemahan ini biasanya catatan child diduplikasi sehingga menciptakan/menyajikan dua hirarki yang terpisah.

GAMBAR MULTIPLE PARENT ASSOCIATION



Sayangnya penyelesaian ini meningkatkan kos pengulangan data. Untuk itu disusun model kedua yaitu:

2) *The Network Model* (gambar 3-12 hal 111)



Model ini sama dengan hirarki, perbedaannya model *network* mengijinkan catatan anak memiliki *parent* lebih dari satu.

3) *The Relation Model*

Daftar nama konsumen

Atribut : nama, alamat, saldo

Cust (Key)	Name	Addres	Current Balance
1875	J Smith	18 Elm St.	1820

Tuples (records)

Perbedaan model *relation* dengan model pertama adalah cara hubungan data disajikan kepada *user.* Model ini menggambarkan data dalam tabel dua

dimensi. Bagian kolom berisi atribut, sedangkan baris berisi kesatuan data yang sama tetapi tidak sama persis, untuk dicatat dalam sistem file *flat*

3. Databases in a Distributed Environment

A. Centralized Databases

Pada Bab 2 telah diperkenalkan konsep dari *Distributed Data Processing* (DDP). Struktur fisik dari data organisasi merupakan pertimbangan yang penting dalam merencanakan suatu sistem pendistribusian. Dalam berbicara isu ini, perencana mempunyai dua opsi dasar: *database* tersebut dapat dipusatkan atau mereka dapat didistribusikan. *Database distributor* dibagi menjadi dua kategori: *database* yang disekat (*partitioned database*) dan *database* yang ditiru (*replicate database*).

Pendekatan pertama melibatkan penahanan data di lokasi pusat. Unit-unit *remote* IT meminta pengiriman data ke lokasi pusat, yang mana proses permintaan-permintaan dan pengiriman data kembali ke permintaan unit IT. Pengolahan sesungguhnya data tersebut dilaksanakan di unit remote IT. Pendekatan *database* yang dipusatkan digambarkan di dalam Gambar 3-15. Suatu tujuan pokok dari pendekatan *database* untuk memelihara peredaran data (currency data). Hal ini merupakan tugas yang menantang di dalam lingkungan DDP.

GAMBAR 3-15 CENTRALIZED DATABASE

GAMBAR 3-16 THE PARTITIONED DATABASE APPROACH

Peredaran data Di dalam Lingkungan DDP

Selama pengolahan data, nilai rekening saldo melewati keadaan tidak konsisten yang dimana sementara nilai-nilai mereka dinyatakan salah. Hal ini terjadi selama eksekusi suatu transaksi.

Untuk mencapai peredaran data, akses bersama ke unsur-unsur data individu dengan unit-unit IT ganda yang harus dicegah. Solusi pada masalah ini adalah menggunakan larangan bekerja database, yang mana adalah suatu kendali perangkat lunak (biasanya suatu fungsi dari DBMS) mencegah akses-akses ganda bersama ke data.

B. Distributed Databases

Dapat berupa *partitioned database* maupun *replicated database*.

1) Partitioned Databases

Pendekatan *partitioned database* memisah database pusat menjadi segmen-segmen atau partisi-partisi yang didistribusikan ke pemakai utama. Keuntungan pendekatan ini:

- Data yang tersimpan pada *site* lokal meningkatkan pengendalian *user.*
- Waktu respon pemrosesan transaksi ditingkatkan dengan mengijinkan akses lokal ke data dan mengurangi volume data yang harus dikirim antara unit IT.
- *Database* yang dipartisi dapat mengurangi efek potensial kerusakan.

a. The Deadlock Phenomenon

Deadlock adalah kondisi permanen yang harus dipecahkan dengan software khusus yang menganalisis setiap kondisi deadlock untuk menentukan solusi terbaik. Pada lingkungan terdistribusi, dimungkinkan bagi site ganda saling mengunci satu sama lain dari database, sehingga mencegah masing-masing dari pemrosesan transaksinya. *Deadlock* terjadi karena ada *mutual exclusion* pada sumber data, dan transaksi ada pada posisi 'menunggu' sampai dengan *lock* dipindah.

b. The Deadlock Resolution

Untuk mengatasi *deadlock* pada umumnya melibatkan lebih dari satu *terminating* atau transaksi untuk melengkapi pemrosesan transaksi lain pada *deadlock.*

GAMBAR 3-17 THE DEADLOCK CONDITION

2) Replicated Databases

Database direplikasi dapat menjadi efektif pada perusahaan tingkat *sharing* datanya tinggi, tidak ada pemakai utama. Pertimbangan utama mereplikasi *database* untuk mendukung *query-query read-only*. Dengan mereplikasi data pada setiap bagian, akses data untuk tujuan query dapat dipastikan, dan *lockuts* dan keterlambatan akibat lalu lintas data dapat diminimalkan. Kelemahan pendekatan ini adalah menjaga (*maintaining*)versi terbaru dari masing-masing *database site.*

GAMBAR 3-18 REPLICATED DATABASE APPROACH

GAMBAR 3-19 REPLCATED DATABASES UPDATED INDEPENDENTLY

C. Concurrency Controls

Database concurrency adalah adanya kelengkapan dan keakuratan data pada semua *data site* pengguna. Perancang sistem memerlukan suatu metode untuk mememastikan bahwa transaksi yang diproses pada setiap *site* tercermin dalam database secara akurat pada seluruh *site.* Permasalahan dalam pengendalian *concurrency* adalah cara/perilaku para auditor. Metoda yang digunakan untuk kendali concurrency adalah transaksi bersambung dengan *time-stamping*. pengecapan. Metoda ini melibatkan label masing-masing transaksi oleh dua kriteria:

1) Software khusus untuk mengelompokkan transaksi ke dalam kelas-kelas untuk mengidentifikasi konflik potensial.
2) Sebagian proses pengendalian adalah *time-stamp* setiap transaksi.

Jika muncul konflik maka transaksi dimasukkan ke dalam *schedule serial* (bersambung).

.

Database Distribution Methods and the Accountant

Keputusan itu untuk mendistribusikan *database* harus penuh pertimbangan, ada beberapa trade-off yang harus dipertimbangkan. Sebagian dari pertanyaan-pertanyaan mendasar untuk diajukan:

- Haruskan data organisasi itu dipusatkan atau didistribusikan?
- Jika distribusi data diinginkan, perlu replikasi database atau partisi data?

- Jika ditiru, perlu database itu secara total atau parsial?
- Jika database dipartisi, bagaimana seharusnya segmen-segmen data dialokasikan di antara lokasi-lokasi?

Pilihan melibatkan di setiap pertanyaan-pertanyaan ini berdampak pada kemampuan organisasi itu untuk memelihara integritas data. Pencegahan jejak audit dan ketelitian dari catatan akuntansi adalah kunci pokok.

4. Controlling and Auditing Data Management Systems

Pengendalian DBMS dikelompokkan menjadi dua yaitu: pengendalian akses (*access control*) dan pengendalian *backup.*

A. Access Controls

Adalah pengendalian yang dirancang untuk mencegah individu tanpa otorisasi menampilkan, memanggil, merusak, dan menghancurkan data entitas.

1) User Views

Adalah subset dari total *database* yang menegaskan domain data pemakai dan menyediakan akses ke *database.*

2) Database Authorization Table

Berisi aturan-aturan yang membatasi tindakan yang dapat diambil pemakai. Teknik ini sama dengan daftar pengendalian akses yang digunakan dalam sistem operasi.

3) User-Defined Procedures

Adalah prosedur yang mengijinkan pemakai untuk menciptakan program keamanan personalatau rutin untuk menciptakan identifikasi pemakai positif lebih daripada *password* tunggal.

4) Data Encryption

Data ecryption digunakan untuk melindungi data yang sifatnya sangat sensitif. Dengan prosedur *data encryption* maka penyusup data tidak dapat membaca data karena database di-*scramble*

5) Biometric Devices

Adalah prosedur yang menggunakan alat biometrik untuk mengukur berbagai macam karakteristik personal, seperti sidik jari.

6) Inference Controls

Salah satu kemampuan *database query* adalah menyediakan ringkasan dan data statistik pengambilan keputusan bagi pengguna.

7) Audit Objectives

Adalah memverifikasi otoritas akses database dan hak pemakai dijamin terkait dengan kebutuhan legitimasi mereka.

8) Audit Procedures:

a. Responsibility for Authority Tables and Subschemas→auditor memverifikasi bahawa petugas DBA bertanggung jawab penuh menciptakan tabel otoritas dan mendesain *user views.*
b. Appropriate Access Authority→auditor memilih sampel user dan memverifikasi hak akses yang disimpan dalam tabe otoritas masih konsisten dengan fungsi organisasional
c. Biometric devices→auditor mengevaluasi kos dan benefit pengendalian biometrik.
d. Inference Controls→auditor memverifikasi pengendalian *database query* yang ada untuk mencegah akses tanpa otorisasi melalui simpulan.

e. Encryption Controls→auditor harus memverifikasi data sensitif.

B. Backup Controls

Adalah pengendalian untuk memastikan bahwa kejadian kehilangan data akibat akses tanpa otorisasis, kegagalan perangkat, atau kerusakan fisik organisasi dapat diperbaiki oleh *database* tersebut.

1) Backup Controls in the Flat-File Environment

Teknik backup yang digunakan tergantung pada media atau struktur *file*.

a. **GPC Backup Technique**

Grant-parent-child backup adalah teknik yang digunakan dalam sistem batch file sekuensial. Prosedur backup dimulai ketika file master sekarang (parents) diproses terhadap file transaksi untuk memproduksi *file* master *updated* (*child).* Pada transaksi *batch* berikutnya, anak menjadi master file, file parent yang asli naik menjadi backup (*grantparent*)

b. Direct Access file Backup

Nilai data pada akses langsung diubah tempatnya dalam suatu proses yang dinamakan destructive replacement. Oleh karena itu, sekali data berubah, nilai asli dihancurkan, dan hanya meninggalkan satu versi terbaru.

c. Off-Site Storage

Adalah tempat penyimpanan backup baik pada GPC maupun pendekatan langsung rorpada tempat yang aman di luar site.

d. Audit Objective

Verifikasi bahwa pengendalian backup cukup efektif dalam melindungi file data dari kerusakan fisik, hilang, kerusakan akibat kegagalan maupun *program error*

e. Audit Procedures

- Sequencetial File (GPC) Backup
- Backup Transactional File
- Direct Access F ile Backup
- Off-site storage

2) Backup Controls in the Database Environment

Pengendalian backup untuk lingkungan *database* menyediakan sistem *backup* dan pemulihan sebagai berikut:

a. Backup

Backup secara periodik untuk seluruh data

b. Transaction Log (Journal)

The transaction log adalah fitur yang menyediakan jejak audit seluruh transaksi yang diproses.

c. Checkpoint Feature

Adalah fasilitas yang menunda seluruh proses sementara data sedang diperbaiki (proses pemulihan data) *transaction log* dan *database* mengubah *log database.*

d. Recovery Module

Modul ini menggunakan logs dan backup untuk memulai kembali setelah sistem mengalami kegagalan

e. Audit Objective

Verifikasi bahwa pengendalianseluruh sumber data cukup untuk menyediakan intergritas dan keamanan *database.*

f. Audit Procedures

Auditor seharusnya memverifikasi backup yang dilakukan secara rutin (sering) untuk mempermudah pemulihan jika ada kehilangan tanpa melakukan proses ulang yang berlebihan.