

## DATABASE DAN USER DATABASE (ref : Fundamentals of DB Systems, Elmasri, N)

Aplikasi tradisional DB : deposit atau penarikan uang, reservasi hotel atau penerbangan, akses katalog perpustakaan yang terkomputerisasi, dll.  
Pada model ini, kebanyakan informasi disimpan dan diakses secara teks atau numeric.

Database multimedia merupakan teknologi lanjut yang dapat menyimpan gambar, video klip, pesan suara. GIS dapat menyimpan dan menganalisa peta, data cuaca, dan citra satelit.

Data warehouse dan OnLine Analytical Processing (OLAP) digunakan untuk mengekstrak dan menganalisis informasi yang berguna dari database yang sangat besar untuk pengambilan keputusan.

Teknologi database aktif dan real-time digunakan untuk mengatur proses industri dan pabrikasi. Teknik pencarian database digunakan di WWW untuk meningkatkan pencarian informasi yang dibutuhkan user.

### PENDAHULUAN

Database dan teknologinya berpengaruh besar terhadap pertumbuhan komputer. Database didefinisikan sebagai kumpulan dari data yang berhubungan. Misalnya nama, nomor telepon, dan alamat yang dapat disimpan (dalam buku, PC, disket).

Sebuah database dapat digenerate atau dimaintain secara manual atau terkomputerisasi. Contoh kartu katalog perpustakaan. Database yang terkomputerisasi data dibuat dan dimaintain oleh program aplikasi yang secara khusus ditulis untuk itu atau oleh sistem manajemen database.

Sistem manajemen database (DBMS) merupakan kumpulan program untuk membuat dan memaintain sebuah database oleh user. DBMS merupakan sistem software general-purpose yang memiliki fasilitas proses define, construct dan manipulate database untuk aplikasi yang bervariasi.

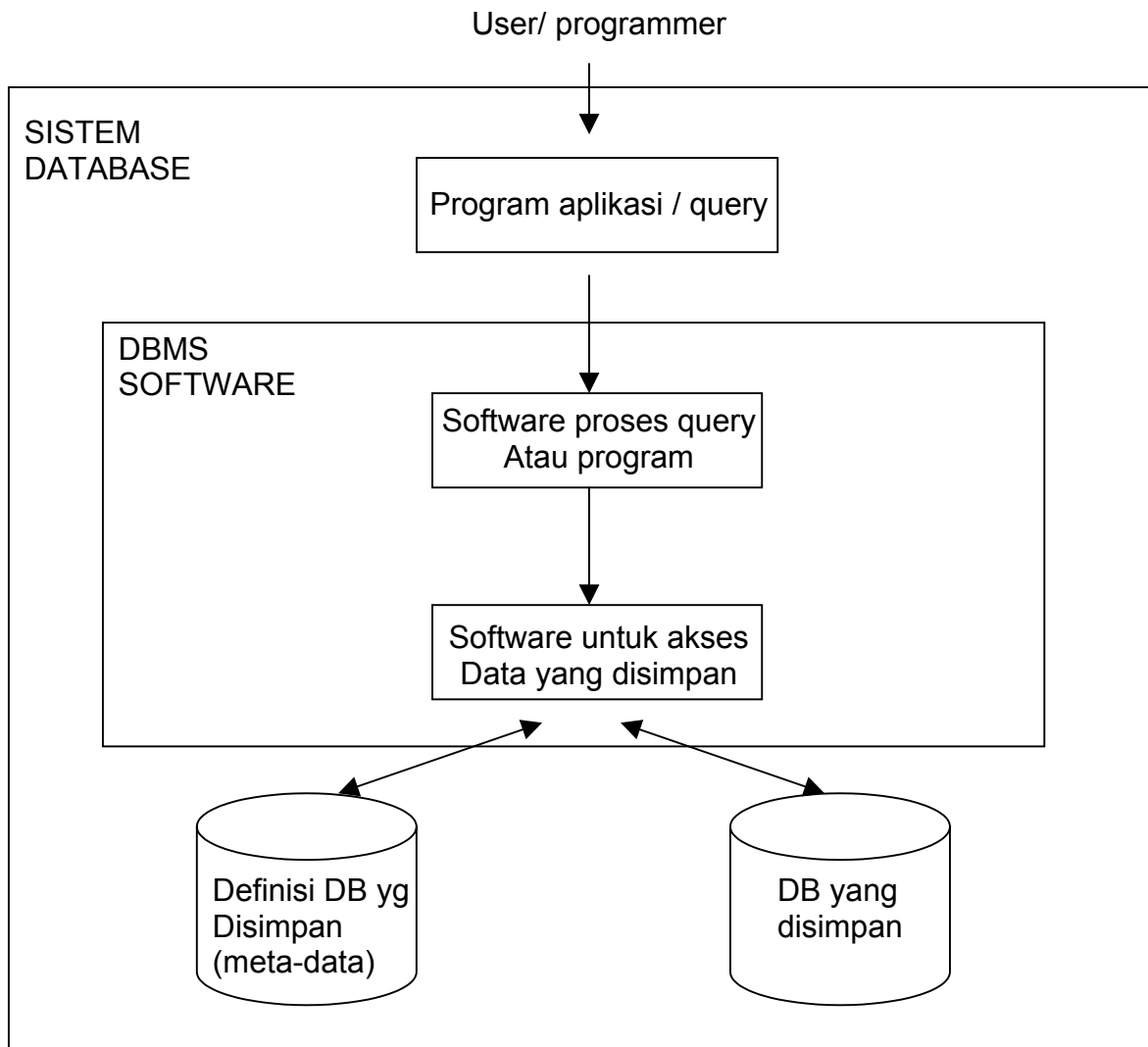
Define : spesifikasi tipe data, struktur dan constraint data yang akan disimpan dalam database

Construct : proses menyimpan data itu sendiri ke dalam beberapa media penyimpanan yang dikontrol DBMS.

Manipulate : fungsi seperti query database untuk memanggil data khusus, update database dan generate laporan dari data.

Software DBMS general-purpose tidak selalu dibutuhkan untuk mengimplementasikan database yang terkomputerisasi. Dapat juga sekumpulan program untuk membuat atau memaintain database, dibuat sendiri (ini yang dinamakan software DBMS special-purpose).

Sistem database merupakan sebutan untuk kedua database dan software DBMS.



Gambar 1. lingkungan sistem database

CONTOH

Database UNIVERSITAS

Database mengorganisasikan 5 buah file masing2 menyimpan record data yang bertipe sama.

STUDENT

Name	StudentNumber	Class	Major
Smith	17	1	CS
Brown	8	2	CS

COURSE

CourseName	CourseNumber	CreditHours	Department
Intro to CS	CS1310	4	CS
Data structure	CS3220	4	CS
Discrete math	MATH2410	3	MATH
Database	CS3380	3	CS

## SECTION

SectionIdentifier	CourseNumber	Semester	Year	Instructor
85	MATH2410	Fall	98	King
92	CS1310	Fall	98	Anderson
102	CS3320	Spring	99	Knuth
112	MATH2410	Fall	99	Chang
119	CS1310	Fall	99	Anderson
135	CS3380	Fall	99	stone

## GRADE\_REPORT

StudentNumber	SectionNumber	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

## PREREQUISITE

CourseNumber	PrerequisiteNumber
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

## DEFINE :

Struktur dari record per file dispesifikasikan dengan tipe elemen data yang berbeda untuk disimpan.

File STUDENT terdiri dari StudentName, StudentNumber, Class dan Major.

Tipe data untuk setiap elemen data dari record juga perlu dibuat.

StudentName merupakan string dari karakter alfabet, StudentNumber dibuat menjadi integer, dst.

Pengkodean juga dibuat, misalnya Class dari STUDENT, 1 untuk freshman, 2 untuk sophomore, 3 untuk junior, 4 untuk senior dan 5 untuk graduate student.

## CONSTRUCT :

Data yang mewakili student, course, section, grade report dan prerequisite disimpan sebagai sebuah record dalam masing2 filenya. Ada beberapa record yang berrelasi. Misalnya record "Smith" di STUDENT berrelasi dengan 2 buah record di file GRADE\_REPORT.

## MANIPULATE :

Berhubungan dengan query dan update.

Query : "retrieve the transcript – a list of all courses and grades – of Smith"; list

Update : "change the class of Smith to Sophomore"; create; enter

Sebelum diproses, query dan update yang informal seperti tersebut perlu diterjemahkan ke bahasa sistem database.

**KARAKTERISTIK DARI PENDEKATAN DATABASE** (dibandingkan dengan proses file yang tradisional)

Dalam pemrosesan file secara tradisional, setiap user mendefinisikan dan mengimplementasikan file yang dibutuhkan untuk aplikasi khusus sebagai bagian dari pemrograman aplikasinya.

Pada pendekatan database, sebuah penyimpanan data dimaintain, sekali didefinisikan kemudian dapat diakses oleh banyak user.

**Karakteristik Utamanya**

1. Self-describing nature of a database system

Sistem database tidak hanya berisi database itu sendiri tetapi definisi atau deskripsi yang lengkap dari struktur dan constraint database. Definisi ini disimpan dalam katalog sistem, yang berisi informasi tentang struktur setiap file, tipe dan format penyimpanan dari item data, dan constraint yang beragam dari data. Informasi yang disimpan dalam katalog ini disebut sebagai meta-data.

Katalog digunakan software DBMS dan juga user database yang membutuhkan informasi tentang struktur dari database. Paket software DBMS general-purpose tidak ditulis untuk aplikasi database khusus, sehingga perlu ada refer ke katalog untuk mengetahui struktur dari file dalam suatu database khusus.

Dalam proses file tradisional, definisi data menjadi bagian dari program aplikasi, sehingga program ini hanya dapat bekerja dengan hanya satu database khusus yang strukturnya dideklarasikan dalam program aplikasi.

“struct”; “class”; “Data Division”

2. Insulation between programs and data, and data abstraction

Dalam proses file secara tradisional, perubahan struktur file akan mengubah semua program yang mengakses file tersebut.

Program akses DBMS tidak membutuhkan perubahan dalam banyak kasus. Struktur dari file data disimpan dalam katalog DBMS secara terpisah dari program pengakses. Ini disebut program-data independence.

Dalam database OO atau object-relational, user dapat mendefinisikan operasi data sebagai bagian dari definisi database. Sebuah operasi (fungsi) dispesifikasikan ke dalam 2 bagian. Interface dari operasi meliputi nama operasi dan tipe data dari argumennya (parameternya). Implementasi atau metode dari operasi dispesifikasikan terpisah dan dapat diubah tanpa berefek terhadap interface. Program aplikasi user dapat beroperasi pada data dengan meminta operasinya melalui nama dan argumennya. Ini disebut program-operation independence.

Kedua istilah independence ini disebut sebagai data abstraction. DBMS memberikan user representasi konseptual data yang tidak mencakup banyak detail bagaimana data disimpan atau bagaimana operasi diimplementasikan.

Model data merupakan tipe dari abstraksi data yang digunakan untuk menyediakan representasi konseptual ini. Model data menggunakan konsep logika seperti objek, property, inter relasi, yang lebih mudah dimengerti user dibandingkan dengan konsep penyimpanan komputer.

Pada pendekatan database, struktur dan organisasi dari masing2 file disimpan dalam katalog. User database melihat dari representasi konseptual file, dan DBMS mengekstrak rincian penyimpanan file dari katalog.

### 3. Support of multiple views of the data

Sebuah VIEW merupakan sub set dari database atau dapat berisi data virtual yang berasal dari file database tetapi tidak disimpan secara eksplisit. DBMS yang multiuser dimana aplikasinya bervariasi perlu fasilitas ini. Beberapa user tidak perlu mengetahui apakah data yang diview-nya disimpan atau tidak.

Contoh : View course prerequisite, cek student yang mengambil semua prerequisite setiap course yang diambilnya.

#### PREREQUISITE

CourseName	CourseNumber	Prerequisites
Database	CS3380	CS3320
		MATH2410
Data structure	CS3320	CS1310

### 4. Sharing of data and multiuser transaction processing

Multiuser DBMS berarti banyak user dapat mengakses database pada saat yang sama. Ini penting jika data untuk multiple applications akan diintegrasikan dan dimaintain dalam sebuah database tunggal. DBMS memerlukan software kontrol konkurensi yang memastikan beberapa user yang meng-update data yang sama berlaku sesuai kontrolnya sehingga hasil yang diupdate benar.

Contoh : saat beberapa petugas pemesanan penerbangan melakukan tugasnya, DBMS harus memastikan bahwa setiap kursi dapat diakses hanya oleh satu petugas di satu saat untuk menentukan penumpang kursi tsb. Tipe aplikasi ini yang disebut sebagai aplikasi OnLine Transaction Processing (OLTP). Aturannya adalah bahwa software multiple database harus memastikan bahwa transaksi yang konkuren beroperasi dengan benar.

## PELAKU YANG TERLIBAT DALAM LINGKUNGAN DATABASE

### 1. Database Administrator

Dalam lingkungan database, sumber utama adalah database itu sendiri dan sumber kedua adalah DBMS dengan software-nya. Pengaturan sumber ini dilakukan oleh seorang Administrator Database (DBA). DBA bertanggungjawab atas otorisasi akses ke database, mnegkoordinir dan memonitor penggunaannya dan mendapatkan sumber hardware dan software yang dibutuhkannya. DBA bertanggungjawab atas masalah2 seperti pelanggaran keamanan atau waktu respon sistem yang buruk. Dalam organisasi yang lebih besar, DBA dibantu oleh seorang staff yang menyelesaikan fungsi2 ini.

## 2. Database designer

Database designer bertanggungjawab atas identifikasi data yang disimpan dalam database dan pemilihan struktur yang sesuai untuk mewakili dan menyimpan data ini. Tugas2 ini perlu dilakukan sebelum database yang sebenarnya diimplementasikan dan berisi data. Selain itu juga bertanggungjawab untuk mengkomunikasikan semua user database untuk memahami kebutuhannya, dan mencapai desain yang sesuai dengan kebutuhan user.

Dalam banyak kasus, desainer adalah seorang staff dari DBA dan kemungkinan ditugaskan untuk hal lain jika desain database selesai dibuat. Desainer database secara khusus berinteraksi dengan setiap kelompok user dan membangun view dari database yang sesuai dengan data dan memproses kebutuhan kelompok tsb. View ini kemudian dianalisis dan diintegrasikan dengan view dari kelompok user yang lain. Desain database akhir mampu mendukung kebutuhan dari semua kelompok user.

## 3. End users

End user merupakan orang2 yang pekerjaannya membutuhkan akses ke database untuk query, update dan generate laporan. Beberapa kategori dari user :

- Casual end user : yang mengakses database, tetapi mereka membutuhkan informasi yang berbeda setiap saat. Mereka menggunakan bahasa query database yang canggih untuk menspesifikasikan permintaan dan mereka adalah manajer tingkat tinggi atau menengah.
- Naïve atau parametric end user : fungsi pekerjaan utama mereka adalah berkisar pada query dan update database, menggunakan tipe standar dari query dan update – disebut canned transaction – yang perlu diprogram dan diuji secara hati2.
- Sophisticated end users : mencakup ahli teknik, ilmuwan, analis bisnis, dan lainnya yang terbiasa dengan fasilitas dari DBMS untuk mengimplementasikan aplikasi sesuai kebutuhannya.
- Stand-alone end users : memaintain database personal dengan menggunakan paket program yang sudah jadi yang menyediakan menu yang easy user dan interface tabg berbasis grafik.

## 4. System analysts and application programmers (software engineers)

Analisis sistem menentukan kebutuhan user khususnya end user yang naïve dan parametric dan membuat spesifikasi untuk canned transaction yang sesuai dengan kebutuhan. Pemrogram aplikasi mengimplementasikan spesifikasi ini sebagai program; kemudian diuji, didebug, didokumentasikan. Software engineers ini perlu terbiasa dengan kemampuan DBMS dalam menyelesaikan tugas2nya.

## 5. Pelaku lainnya

- DBMS system designers and implementers
- Tools developers : orang2 yang mendesain dan mengimplementasikan tool – paket software yang menyediakan dan menggunakan desain sistem database dan meningkatkan kinerja.
- Operators and maintenance personnel : bertanggungjawab atas hardware dan software dari sistem database yang dioperasikan dan dimaintenace

## KEUNTUNGAN MENGGUNAKAN DBMS

### 1. Controlling redundancy

Redundansi terjadi jika banyak data disimpan dua kali, masing2 pada file dari setiap kelompok user. Kelompok user yang lain boleh menyalin beberapa atau semua data yang sama ke file mereka. Beberapa masalah yang timbul yaitu pertama kebutuhan untuk update secara logika menjadi berulang2. Kedua adalah ruang penyimpanan yang besar ketika data yang sama disimpan berulang2. File yang berisi data yang sama, menjadi tidak konsisten. Meskipun update diaplikasikan ke seluruh file yang sesuai, data tetap tidak konsisten karena update dilakukan bebas oleh setiap kelompok user.

Dalam pendekatan database, view dari kelompok user yang berbeda diintegrasikan selama desain database. Untuk konsistensi, perlu desain database yang menyimpan setiap item data logika dalam hanya satu lokasi pada database. Dengan redudansi yang terkontrol memungkinkan kinerja dari query meningkat.

CONTOH

#### GRADE\_REPORT (a)

StudentNumber	StudentName	SectionIdentifier	CourseNumber	Grade
17	Smith	112	MATH2410	B
17	Smith	119	CS1310	C
8	Brown	85	MATH2410	A
8	Brown	92	CS1310	A
8	Brown	102	CS3320	B
8	Brown	135	CS3380	A

(a) Controlled redundancy : ada StudentName dan CourseNumber-nya.

#### GRADE\_REPORT (b)

StudentNumber	StudentName	SectionIdentifier	CourseNumber	Grade
17	Brown	112	MATH2410	B

(b) Uncontrolled redundancy : tidak konsisten, karena nama mahasiswa 17 adalah Smith, bukan Brown.

### 2. Restricting unauthorized access

Ketika multiple users berbagi database, ada beberapa user yang tidak diotorisasikan untuk mengakses semua informasi dari database.

Beberapa ini mungkin diijinkan untuk retrieve data, meskipun yang lainnya diijinkan untuk retrieve dan update. Operasi retrieve dan update juga perlu dikontrol. Secara khusus user atau kelompok user memiliki password untuk dapat mengakses database.

DBMS menyediakan keamanan dan subsistem otorisasi yang DBA gunakan untuk membuat account dengan batasan2nya.

### 3. Providing persistent storage for program object and data structures

Ini yang mengawali sistem database berorientasi objek. Misal tipe record dalam pascal atau definisi kelas di C++. Nilai dari variable program dihilangkan setiap program selesai, kecuali pemrogram menyimpannya secara permanen dalam file, yang biasanya dikonversi ke format yang sesuai. Untuk membacanya, pemrogram

harus mengkonversi dari format file ke struktur variable program. Objek ini disebut persistence.

Struktur data pada tradisional database tidak kompatibel dengan struktur data bahasa pemrograman, disebut impedance mismatch problem.

#### 4. Permitting inferencing and actions using rules

Sistem database deduktif memiliki kemampuan mendefinisikan rule deduksi untuk menginfer informasi baru.

Misal menentukan siswa dalam masa percobaan. Ini dideklarasikan sebagai rule. Pada DBMS tradisional, kode program prosedural seperti ini secara eksplisit perlu ditulis. Tetapi jika rule diubah, yang tepat diubah adalah rule deduksi yang dideklarasikan daripada mengkode prosedur programnya.

Sistem database aktif menyediakan rule yang aktif yang dapat secara otomatis menginisialisasi aksi.

#### 5. Providing multiple user interfaces

Karena banyak tipe user dengan level pengetahuan teknik yang bermacam2 dalam menggunakan database, DBMS perlu menyediakan interface user yang bermacam2 pula, yaitu bahasa query bagi casual user; bahasa pemrograman interface untuk pemrogram aplikasi; form dan kode perintah bagi parametric user; menu-driven interface dan natural-language interface bagi stand-alone user (dikenal dengan GUI).

#### 6. Representing complex relationships among data

Database terdiri dari bermacam2 data yang saling berhubungan. DBMS memiliki kemampuan untuk mewakili bermacam2 hubungan yang kompleks diantara data secara mudah dan efisien.

#### 7. Enforcing integrity constraints

DBMS memiliki kemampuan untuk membuat suatu integrity constraint. Tipe yang paling sederhana dari integrity constraint adalah menspesifikasikan tipe data untuk setiap item data. Misal item data untuk Class yang boleh disimpan adalah integer 1 hingga 5, nilai Name harus string dan tidak lebih dari 30 karakter.

Perancang database bertanggungjawab untuk mengidentifikasi integrity constraint selama perancangan database. Beberapa constraint dibuat dan secara otomatis dijalankan, beberapa lagi ada yang harus diperiksa dengan program update atau pada saat data entry.

#### 8. Providing backup and recovery

Backup dan sub sistem recovery merupakan fasilitas yang harus disediakan DBMS. Misal jika sistem komputer gagal saat sedang mengupdate program, sub sistem recovery bertanggungjawab untuk memperbaiki atau memastikan database direstore ke keadaan sebelum program dieksekusi kembali. Atau sub sistem recovery memastikan bahwa program diresume dari keadaan dimana diinterupsi sehingga database dapat menyimpannya.

## IMPLIKASI DARI PENDEKATAN DATABASE

### *Potential for enforcing standards*

Fasilitas ini menyediakan DBA untuk dapat mengkomunikasikan dan membuat kerjasama diantara banyak bagian, proyek atau user dalam organisasi. Standard dibuat untuk nama dan format elemen data, tampilan format, struktur laporan, terminology, dst. DBA dapat dengan mudah membuat standard ini dalam database sentral dibandingkan masing2 kelompok membuat file dan softwarenya sendiri.

### *Reduced application development time*

Mendesain dan mengimplementasikan database baru dari awal memakan waktu lebih lama dibandingkan menulis aplikasi file khusus. Tetapi, sekali database dibuat dan dijalankan, waktu yang dibutuhkan untuk membuat aplikasi baru menjadi semakin cepat dengan fasilitas database. Waktu membangun dengan DBMS diperkirakan 1/6 atau 1/4 dari sistem file tradisional.

### *Flexibility*

Struktur sewaktu2 dapat diubah sesuai permintaan. DBMS yang modern dapat mengubah beberapa tipe tertentu ke struktur database tanpa mengganggu data yang disimpan atau program aplikasi yang ada.

### *Availability of up-to-date information*

Ketika seorang user mengupdate database, semua user lainnya dapat melihat perubahan tersebut. Informasi yang uptodate menjadi penting dalam aplikasi proses transaksi. Misal sistem reservasi atau database perbankan.

### *Economics of scale*

Gabungan dari data dan aplikasi akan mengurangi ketumpangtindihan (overlap) dari kegiatan proses data perorangan di berbagai proyek atau bagian. Sehingga dapat membuat organisasi menginvestasikan ke hardware dan software secara menyeluruh dibandingkan untuk masing2 bagian organisasi. Ini juga akan mengurangi biaya operasi dan manajemen.

## KAPAN TIDAK MENGGUNAKAN DBMS

Biaya overhead dari penggunaan database terjadi jika :

- Investasi awal dari hardware, software dan pelatihan tinggi
- Mendefinisikan dan memproses data
- Overhead untuk keamanan, kontrol konkurensi, recovery dan fungsi integrity

Masalah akan bertambah jika perancang database dan DBA tidak sesuai merancang database atau jika aplikasi sistem database tidak diimplementasikan dengan baik.

Akan lebih baik menggunakan model konvensional atau tradisional jika :

- Database dan aplikasinya sederhana, mudah didefinisikan dan tidak mengalami perubahan,
- Ada beberapa kebutuhan real-time yang mendesak di beberapa program yang tidak sesuai karena overhead dari DBMS
- Multiple-user access ke data tidak diperlukan.