Istilah-istilah dalam sistem database :

- ***Enterprise***, merupakan suatu bentuk organisasi, seperti : instansi, sekolah, bank, pabrik, dan lain-lain.
- ***Entitas***, suatu obyek yang dapat dibedakan dengan obyek lainnya yang dapat diwujudkan dalam database.

Contoh :

Entitas dilingkungan universitas → mahasiswa, dosen, matakuliah

Entitas dilingkungan instansi pemerintah → karyawan, departement

- ***Atribute/Field,*** merupakan karakteristik entitas tertentu.

Contoh :

Entity Mahasiswa → atributenya adalah NPM, Nama, Alamat dan lain-lain.

Entity Bank → atributenya adalah Nomor_Rekening, Nama_Nasabah.

- ***Data Value,*** merupakan data actual atau informasi yang disimpan ditiap data elemen atau atribute. Isi dari atribute disebut nilai data (data value).

Contoh :

- ✓ Atribute Nama_Mahasiswa → Wayan Leger, Ketut Doni, Antony.
- ✓ Atribue Alamat_Mahasiswa → Banjar Tegal, Banyuning, Penarukan.

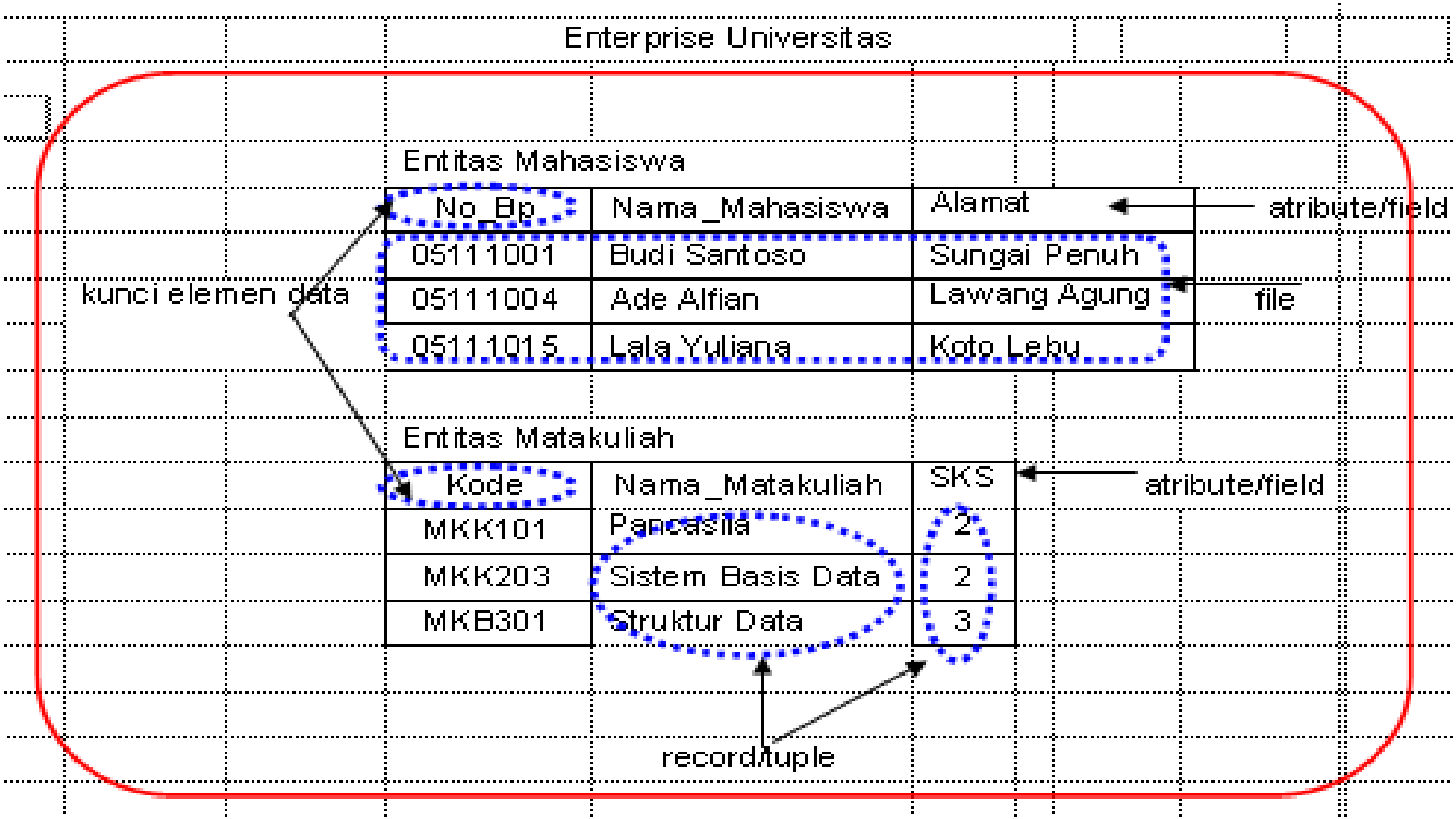
- ***Record/Tuple,*** merupakan kumpulan isi elemen data yang saling berhubungan menginformasikan tentang suatu entity secara lengkap.

Contoh :

- ✓ Kumpulan atribute Nomor_Bp, Nama_Mahasiswa dan Alamat berisikan “05111001”, Wayan Leger, Banjar Tegal.
- ✓ Kumpulan atribute Kode_Obat, Nama_Obat berisikan K01, Konidin.

- ***File,*** kumpulan record sejenis yang mempunyai panjang elemen dan atribute yang sama.
- ***Kunci Elemen Data,*** merupakan tanda pengenal yang secara unik mengidentifikasikan entitas dari suatu kumpulan entitas.
- ***Database Management System (DBMS),*** merupakan kumpulan file yang saling berkaitan bersama dengan program untuk pengelolanya atau merupakan kumpulan software yang mengkoordinasikan semua kegiatan yang berhubungan dengan database agar dapat diakses/dipakai oleh user (pemakai).

Dari pengertian yang dijelaskan diatas secara umum dapat digambarkan sebagai berikut :



Kegunaan Database

Penyusunan database digunakan untuk mengatasi masalah-masalah pada penyusunan data yaitu :

a. Redudansi Data

Redudansi data adalah munculnya data-data yang sama secara berulang-ulang pada file data gabungan yang semestinya tidak perlu terjadi.

Data redudansi perlu dihindari karena :

- Mengakibatkan pemborosan penggunaan media penyimpanan
- Proses updating yang lebih lama
- Terjadinya ketidak konsistensi data yang semakin besar

Contoh :

Struktur File Karyawan

NIP	Nama Karyawan	Golongan	Gaji Pokok
002111045	Enti Gustina	II	650.000,-
002111025	Yetty Nurwati	III	750.000,-
002111042	M. Fajri	II	650.000,-
002111023	Edwar	II	650.000,-

Dari contoh diatas dapat dilihat terjadinya redudancy data yakni pada field golongan dan gaji pokok. Dimana setiap kali rincian record golongan dimasukkan maka akan muncul pula rincian data gaji pokok. Sehingga kerangkapan data akan terjadi pada file tersebut.

b. Inkonsistensi Data

Inkonsistensi Data terjadi akibat :

* Kesalahan dalam pemasukan data (data entry)
* Update anomaly, yaitu : proses untuk mengapdate data tetapi mengakibatkan munculnya data yang tidak konsisten atau kehilangan informasi obyek yang ditinjau.

Contoh :

Inkonsistensi Data dalam File Mahasiswa dan KRS :

File Mahasiswa

NPM	Nama Mahasiswa	Alamat	Tgl_Lahir
03111001	Kurniadi	Sungai Penuh	05-12-1985
03111005	Linda Yanti	Semurup	06-02-1986

File KRS :

NPM	Nama Mahasiswa	Jml_Matakuliah	Jml_SKS
03111001	Kurniadi	3	7
03111005	Linda Yuliana	7	18

Pada contoh diatas terjadi ketidak konsistensi data pada field nama mahasiswa dimana pada record NPM “03111005” yang seharusnya record nama mahasiswa pada file KRS tertulis Linda Yanti, karena terjadi kesalahan entry data tertulis Linda Yuliana.

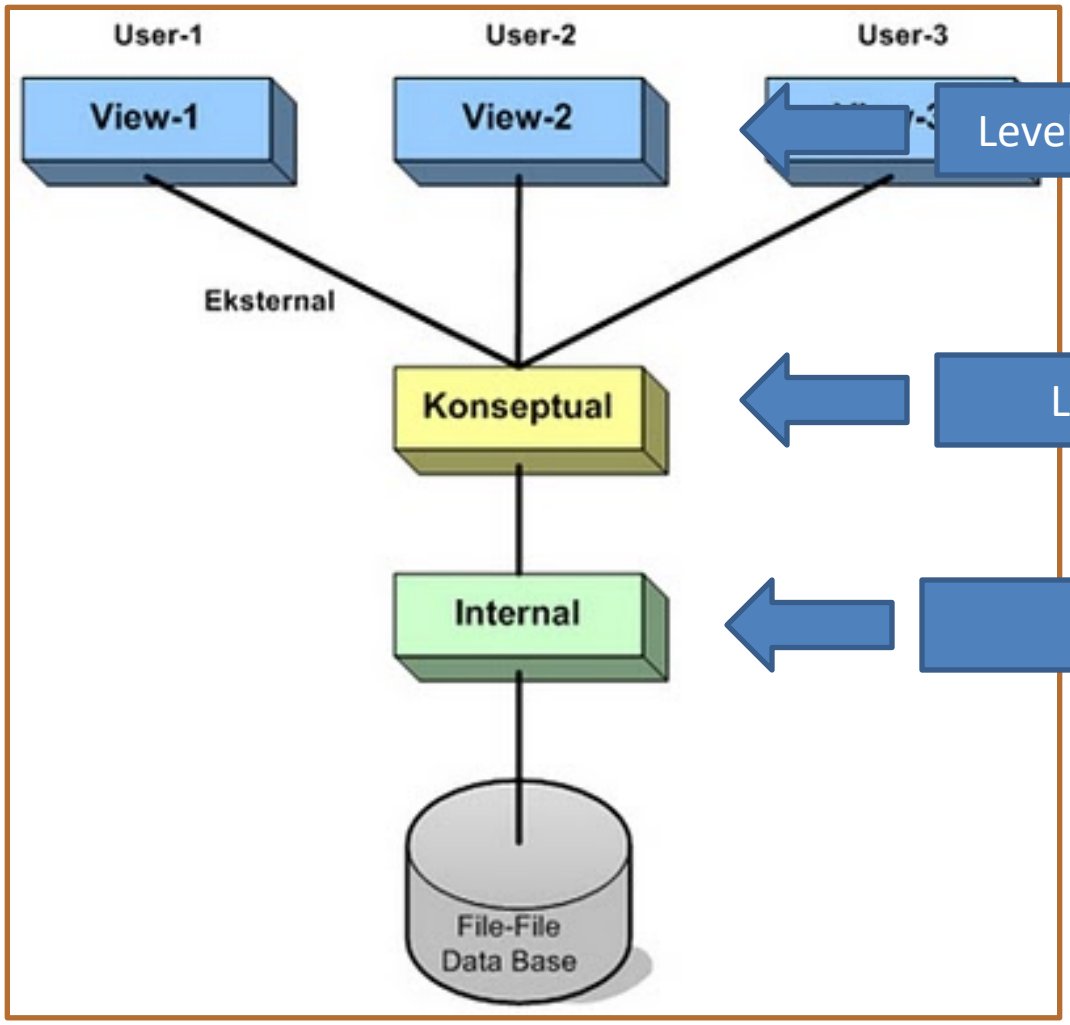
Data Inkonsistensi perlu dihindari karena akan mengakibatkan kesalahan yang fatal pada hasil pengolahan database yang tidak sesuai dengan fakta atau kenyataan yang ada.

ARSITEKTUR BASIS DATA

Pada tahun 1975, badan standarisasi nasional Amerika ANSI-SPARC (American National Standards Institute – Standards Planning and Requirements Committee) menetapkan tiga level abstraksi dalam database, yaitu:

1. Level Eksternal (*external level*) atau Level Pandangan (*view level*)
2. Level Konseptual (*conceptual level*)
3. Level Internal (*internal level*) atau Level Fisik (*physical level*)

ARSITEKTUR BASIS DATA



ARSITEKTUR BASIS DATA

External Level atau View Level

- Level Eksternal adalah level yang berhubungan langsung dengan pengguna database.
- Pada level ini pengguna (user) hanya bisa melihat struktur data sesuai dengan keperluannya sehingga setiap user bisa memiliki pandangan (view) yang berbeda dari user lainnya.

- Pada level ini pula dimungkinkan pandangan user berbeda dengan representasi fisik dari data, misalkan untuk data hari secara fisik data direkam dalam bentuk kode (1, 2, 3, dst) sedang user melihat data dalam bentuk teks nama hari (Ahad, Senin, Selasa, …).
- Data yang dilihat oleh user seakan-akan berasal dari satu file, secara fisik mungkin diambil dari beberapa file yang berelasi.

ARSITEKTUR BASIS DATA

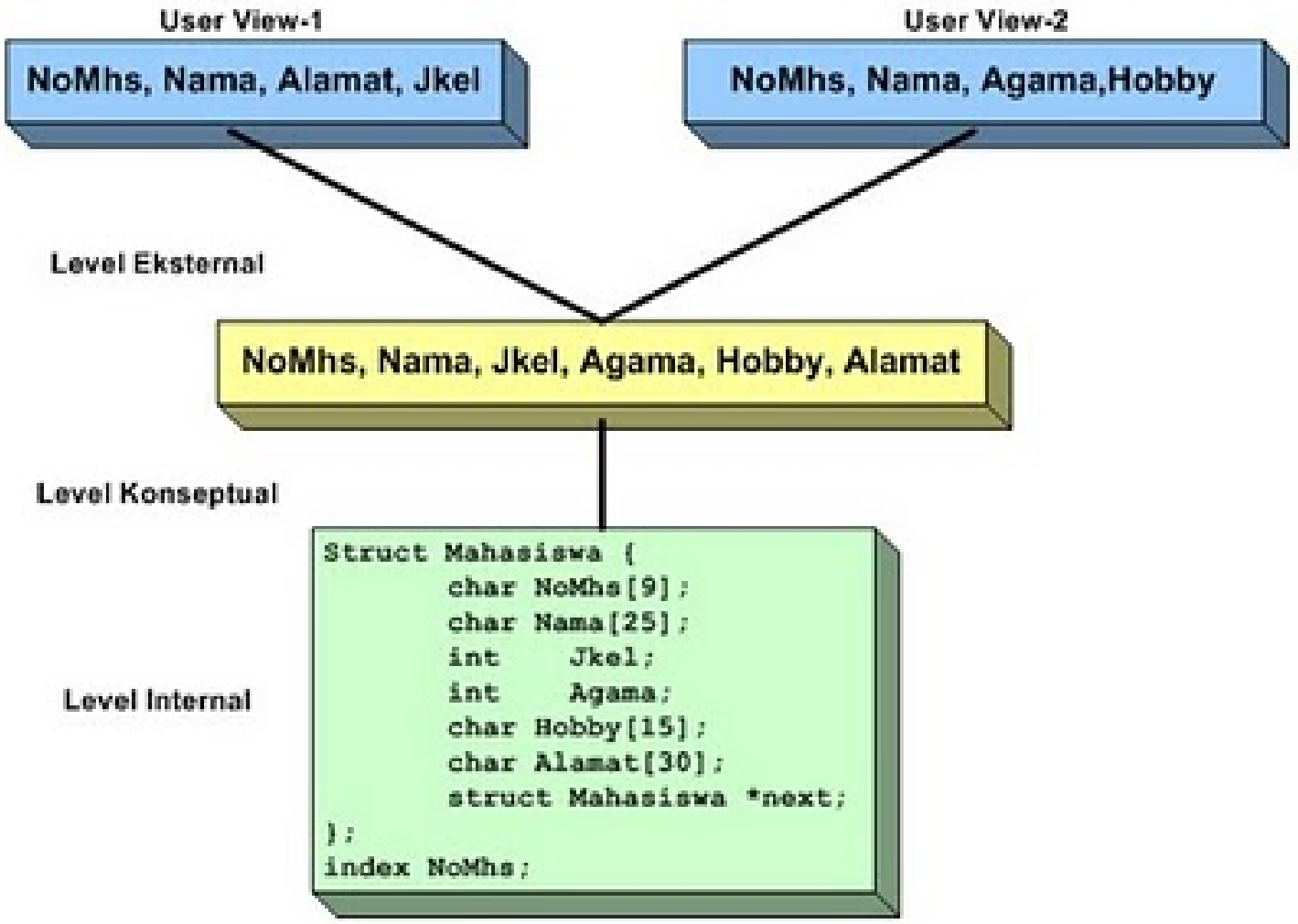
Level Konseptual

- Level Konseptual adalah level dari para administrator database, pada level ini didefinisikan hubungan antar data secara logik, sehingga diperlukan struktur data secara lengkap.
- Para administrator database memahami bagaimana satu view dijabarkan dari beberapa file data, demikian pula pada saat perancangan database mereka dapat saja membagi data menjadi beberapa file agar dapat diakses dan disimpan secara efisien.

ARSITEKTUR BASIS DATA

Level Internal

- Level Internal adalah level dimana data disimpan secara fisik dalam bentuk kode, teks, angka, bit.
- Pada level ini didefinisikan allokasi ruang penyimpanan data, deskripsi data dalam penyimpanan, kompressi data (agar lebih hemat), dan enkripsi data (agar lebih aman).


User View-1
NoMhs, Nama, Alamat, Jkel
User View-2
NoMhs, Nama, Agama,Hobby
Level Eksternal
NoMhs, Nama, Jkel, Agama, Hobby, Alamat
Level Konseptual
Level Internal
Struct Mahasiswa {
char NoMhs[9];
char Nama[25];
int Jkel;
int Agama;
char Hobby[15];
char Alamat[30];
struct Mahasiswa *next;
};
index NoMhs;

Konsep dari level-level tersebut akan menambah pengertian mengenai kebebasan data/data independence.

Data independence dapat dibagi menjadi dua bagian :
1. Physical data independence.
2. Logical data independence.

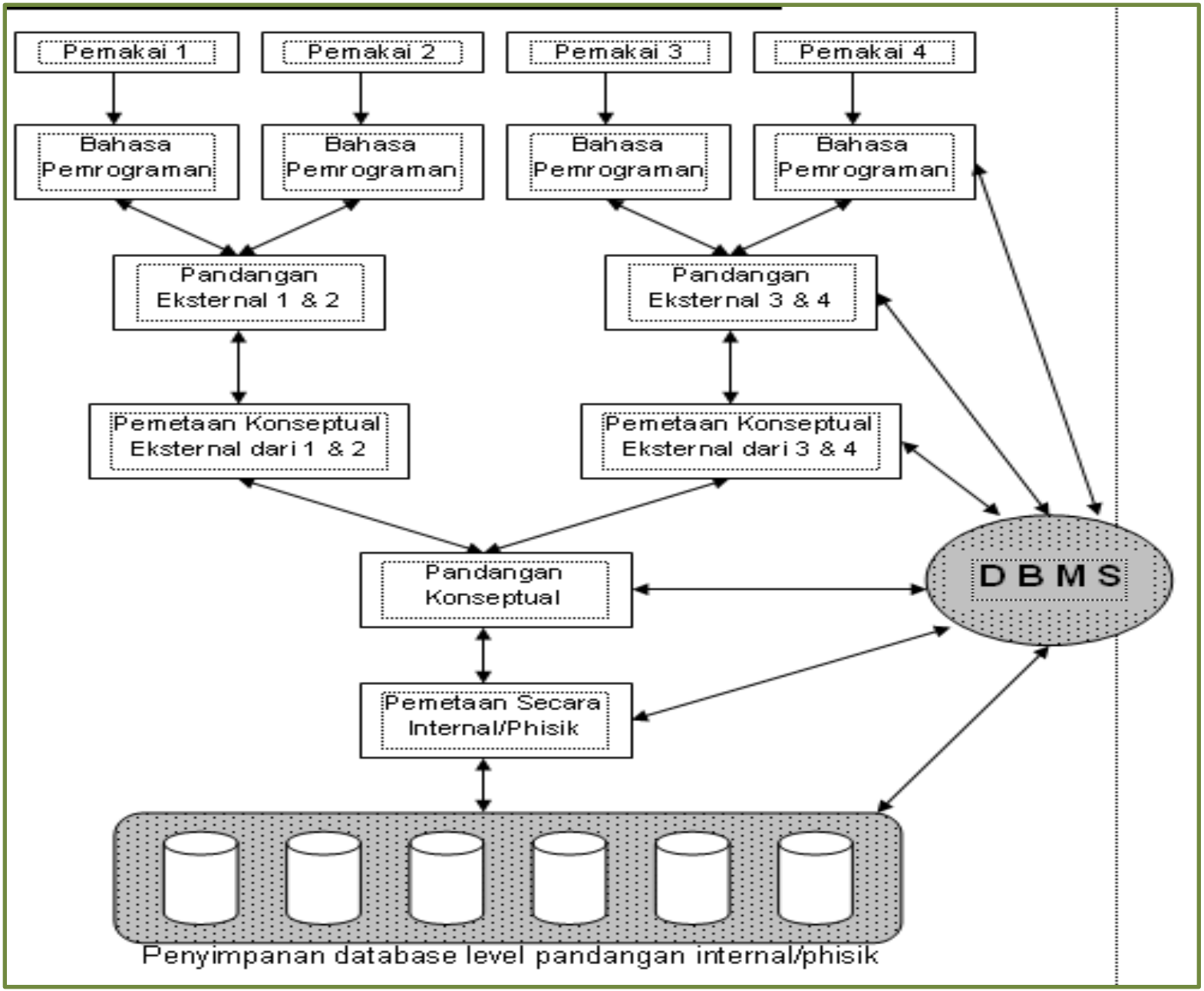
Physical Data Independence:

Kebolehan untuk mengubah pola fisik database tanpa mengakibatkan suatu aplikasi program ditulis kembali. Modifikasi pada level fisik biasanya pada saat meningkatkan daya guna.

Logical Data Independence

Kebolehan untuk mengubah pola konseptual tanpa mengakibatkan suatu aplikasi program ditulis kembali. Modifikasi pada level ini khusus saat struktur logika database ditambahkan/dikurangi.

Agar independensi data dapat dicapai maka disediakan pemetaan antar lapisan (level), yaitu ***pemetaan eksternal-konseptual*** dan ***pemetaan konseptual-internal***.


Pemakai 1
Pemakai 2
Pemakai 3
Pemakai 4
Bahasa Pemrograman
Bahasa Pemrograman
Bahasa Pemrograman
Bahasa Pemrograman
Pandangan Eksternal 1 & 2
Pandangan Eksternal 3 & 4
Pemetaan Konseptual Eksternal dari 1 & 2
Pemetaan Konseptual Eksternal dari 3 & 4
Pandangan Konseptual
DBMS
Pemetaan Secara Internal/Phisik
Penyimpanan database level pandangan internal/phisik

Alasan perlunya prinsip data independence diterapkan pengelolaan sistem database adalah :

1. Database administrator dapat mengubah isi, lokasi, dan organisasi database, tanpa mengganggu program aplikasi yang sudah ada.
2. Vendor hardware dan software pengelolaan data bisa memperkenalkan produk-produk baru, tanpa mengganggu program-program aplikasi yang sudah ada.
3. Memudahkan perkembangan program aplikasi.
4. Memberikan fasilitas pengontrolan terpusat oleh DBA demi security dan integritas data dengan memperhatikan perubahan-perubahan kebutuhan user (pemakai).

MODEL DATA

Model Data :

Kumpulan perangkat konseptual untuk menggambarkan data, hubungan data, semantik data dan batasan data.

Beberapa Jenis Model Data :

1. Model data File datar (Flat-file data model)
2. Model data Hirarki (Hierarchichal data model)
3. Model data Jaringan (Network data model)
4. Model data Relasional (Relational data model)
5. Model data Keterhubungan Entitas (Entity Relationship data model)
6. Model data Berorientasi Objek (Object Oriented data model)

MODEL DATA

MODEL DATA FILE DATAR (FLAT-FILE DATA MODEL)

a. Data flat-file terdiri dari satu atau lebih file yang dapat dibaca, yang secara normal berbentuk format file text.
b. Informasi pada suatu flat-file disimpan sebagai fields, dengan fields-nya memiliki panjang konstan atau panjang bervariasi yang dipisahkan beberapa karakter (delimeter).

MODEL DATA

Contoh 1, Flat-file Model Data

Model data flat-file dengan panjang fields-nya konstan.

1234	5	678901234567890123 45	6	7890123456789012 3
0123		Mulyono		Progdi TI-S1
1234		Arifin		Progdi TI-S1
2345		Tyas Catur P.		Progdi TI-S1
3456		Ifan Riska		Progdi TI-S1
4567		Ayu Pertiwi		Progdi TI-S1

MODEL DATA

Penjelasan Contoh 1,

- Terdapat 3 fields : identifikasi angka, nama dosen, dan nama program studi.
- Setiap fields memiliki panjang konstan karena field identifikasi angka selalu dimulai pada kolom #1 dan selalu berakhir pada kolom #4, field nama dosen selalu dimulai pada kolom #6 dan selalu berakhir pada kolom #25, dan seterusnya.

MODEL DATA

Contoh 2, Flat-file Model Data

Model data flat-file dengan panjang fields-nya bervariasi

0123: Mulyono: Progdi TI-S1
1234: Max Tetelepta : Progdi TI-S1
2345: Tyas Catur P.: Progdi TI-S1
3456: Ifan Riska: Progdi TI-S1
4567: Ayu Pertiwi: Progdi TI-S1
5678: Etika Kartika: Progdi TI-S1
6789: Anthoni Suteja: Progdi TI-S1
7890: Fikri Budiman: Progdi TI-S1

MODEL DATA

Penjelasan Contoh 2,

- Model data flat-file dengan panjang fields bervariasi yang dipisahkan dengan delimeter.
- Untuk setiap fields dipisahkan dengan titik dua. Setiap fields memiliki panjang tidak konstan.
- Pada saat menggunakan fields separator, seharusnya fields seperatornya bukan merupakan karakter yang terdapat pada data.

MODEL DATA

Kelemahan model data flat-file:

- Flat-file tidak menggunakan struktur data yang dengan mudah dapat direlasikan
- Sulit untuk mengatur data secara efisien dan menjamin akurasi
- Lokasi fisik fields data dengan file harus diketahui
- Program harus dikembangkan untuk mengatur data

MODEL DATA

MODEL DATA HIRARKI (HIERARCHICHAL DATA MODEL)

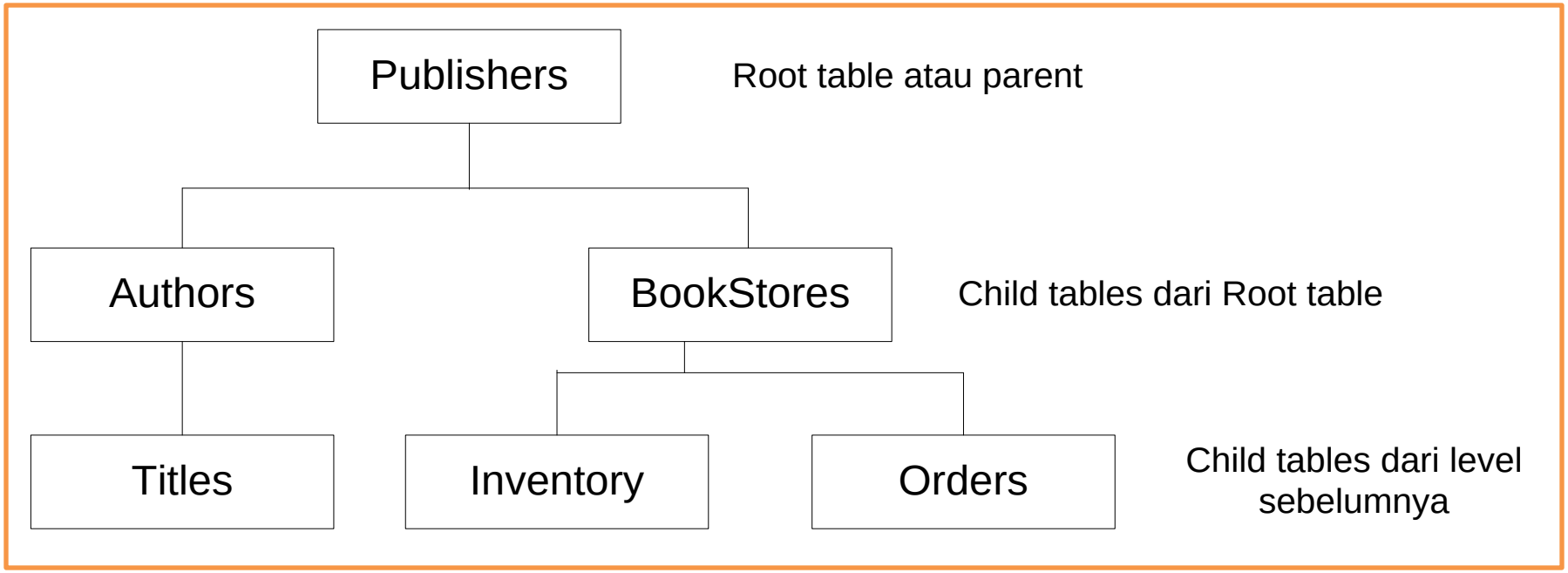
a. Basis data Hirarki satu tingkat di atas basis data flat-file, dalam hal ini kaitanya dengan kemampuan untuk menemukan dan memelihara relasi antar kelompok data

b. Arsitektur model data hirarki berdasarkan konsep hubungan parent/child

c. Pada model data hirarki, suatu root table atau parent table berada apa struktur yang paling atas, terhubung ke child table yang dihubungkan dengan data

MODEL DATA

Contoh: Hirarki Model Data



MODEL DATA

Keterangan :

Kelebihan basis data hirarki dibandingkan flat-file:

- ❑Data dapat dengan cepat dilakukan retrieve
- ❑Integritas data mudah dilakukan pengaturan

Kelemahan basis data hirarki dibandingkan flat-file:

- ❑Pengguna harus sangat familiar dengan struktur basis data
- ❑Terjadi redudansi data

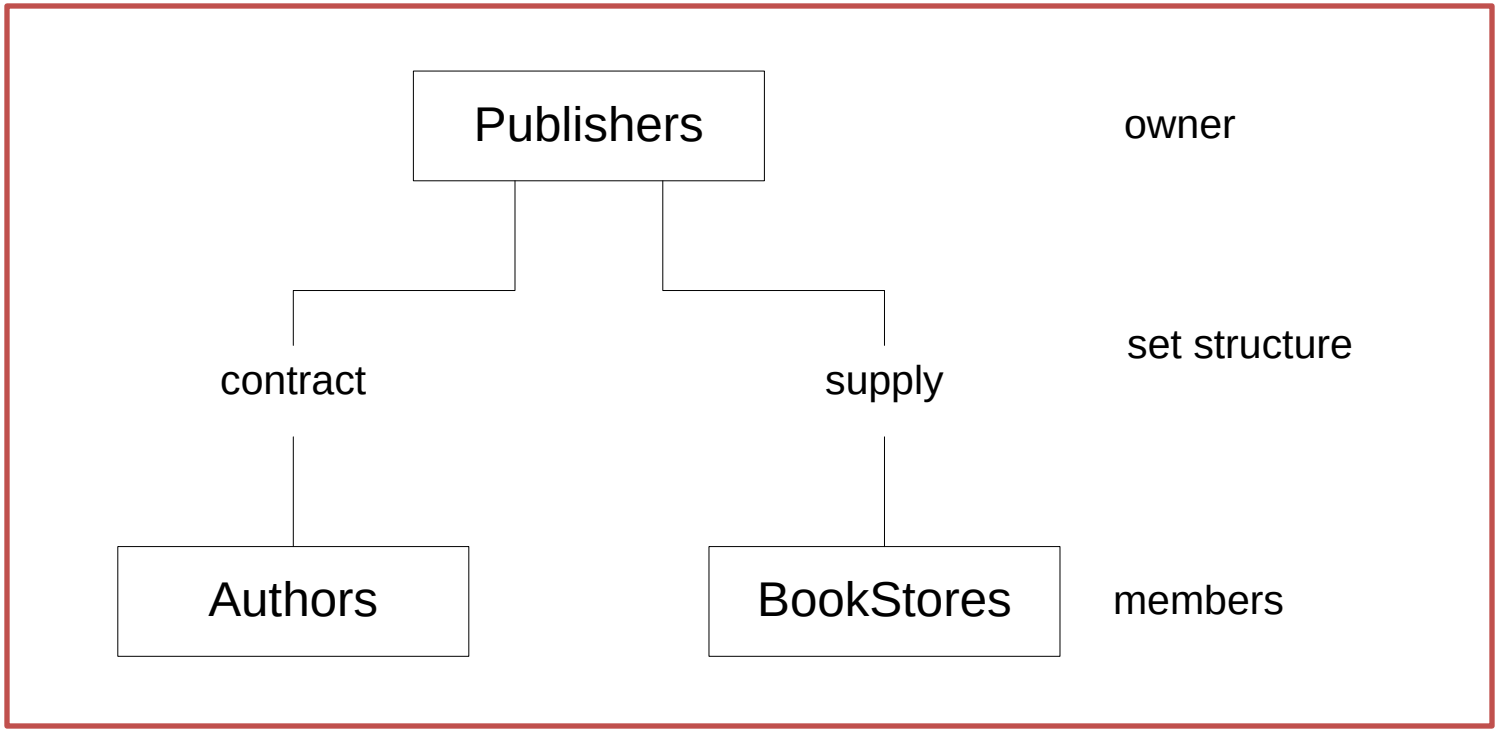
MODEL DATA

MODEL DATA JARINGAN (NETWORK DATA MODEL)

- Model basis data jaringan merupakan perbaikan dari model basis data hirarki, yaitu dengan menambahkan kemampuan root table untuk melakukan share relationships dengan child tables.
- Dalam hal ini child table dapat memiliki banyak root table dan untuk melakukan akses terhadap child table, tidak dibutuhkan lagi untuk mengakses root table terlebih dahulu.

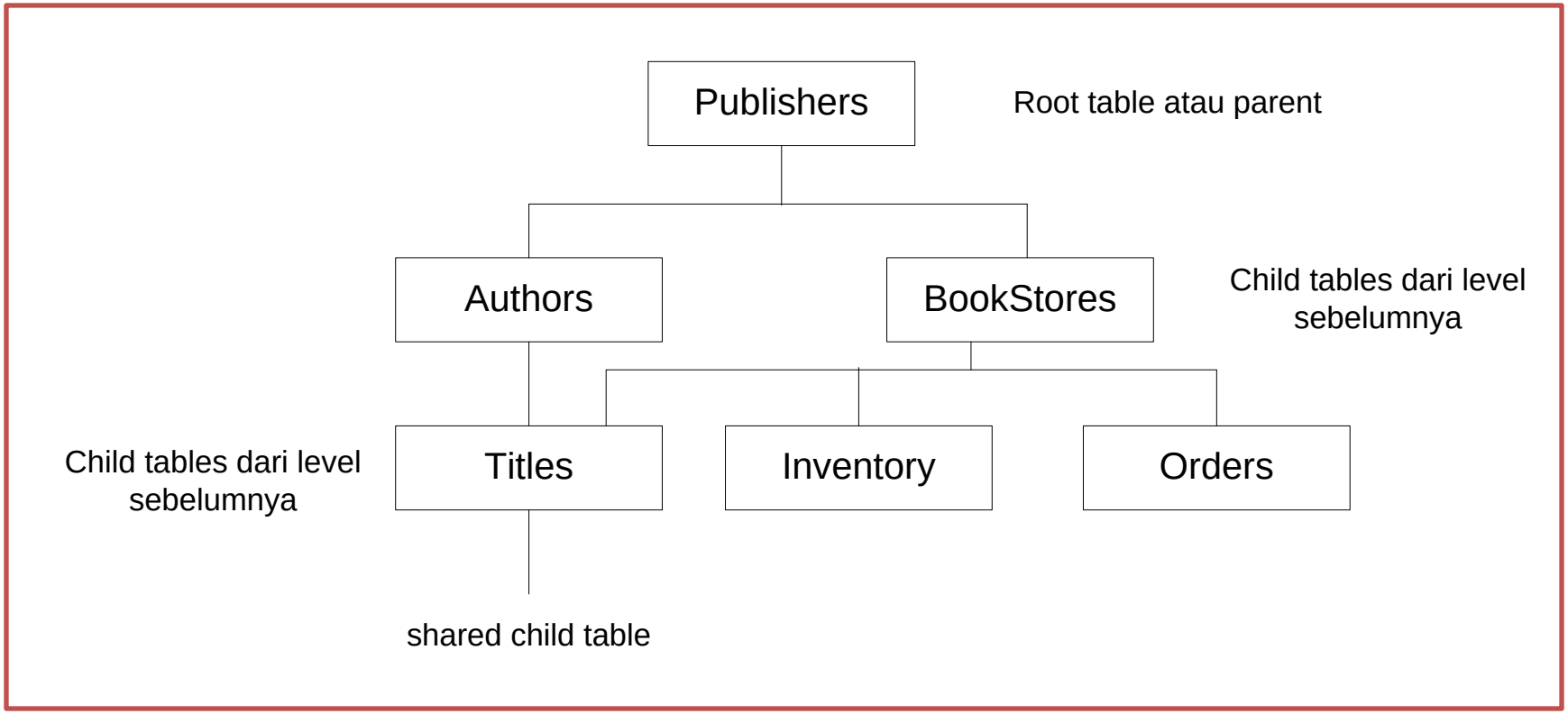
MODEL DATA

Contoh 1, Model Data Jaringan



MODEL DATA

Contoh 2, Model Data Jaringan



MODEL DATA

Model Data Jaringan

Kelebihan model data jaringan:

- Data lebih cepat diakses
- User dapat mengakses data dimulai dari beberapa tabel
- Mudah untuk memodelkan basis data yang komplek
- Mudah untuk membentuk query yang komplek dalam melakukan retrieve data.

MODEL DATA

Model Data Jaringan

Kelemahan basis data jaringan:

- Struktur basis datanya tidak mudah untuk dilakukan modifikasi
- Perubahan struktur basis data yang telah didefinisikan akan mempengaruhi program aplikasi yang mengakses basis data
- User harus memahami struktur basis data.

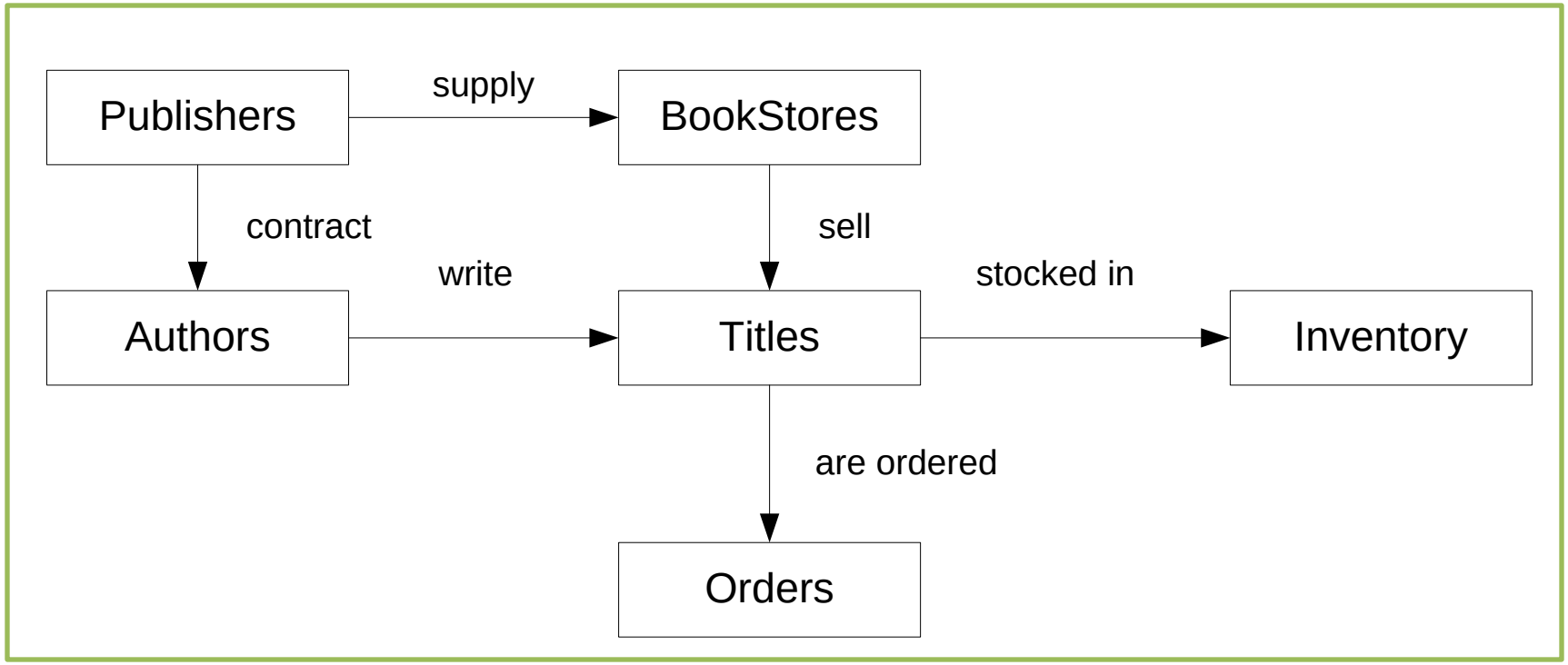
MODEL DATA

MODEL DATA RELASIONAL (RELATIONAL DATA MODEL)

- Model basis data relasional merupakan model basis data yang paling populer banyak digunakan sekarang ini
- Unit utama yang disimpan pada basis data adalah berbentuk tabel atau kelompok data yang saling berhubungan
- Tabel terdiri baris dan kolom, baris adalah merepresentasikan tuple atau record pada tabel, dan kolom merepresentaksikan fields pada tabel
- Tabel dapat berhubungan dengan tabel yang lain dengan menggunakan kunci

MODEL DATA

Contoh : Model Data Relasional



MODEL DATA

Kelebihan basis data relasional:

a. Data sangat cepat diakses
b. Struktur basis data mudah dilakukan perubahan
c. Data direpresentasikan secara logik, user tidak membutuhkan bagaimana data disimpan.
d. Mudah untuk membentuk query yang komplek dalam melakukan retrieve data
e. Mudah untuk mengimplementasikan integritas data
f. Data lebih akurat
g. Mudah untuk membangun dan memodifikasi program aplikasi
h. Telah dikembangkan Structure Query Language (SQL).

MODEL DATA

Kelemahan basis data relasional:

a. Kelompok informasi/tables yang berbeda harus dilakukan joined untuk melakukan retrieve data
b. User harus familiar dengan relasi antar tabel
c. User harus belajar SQL.

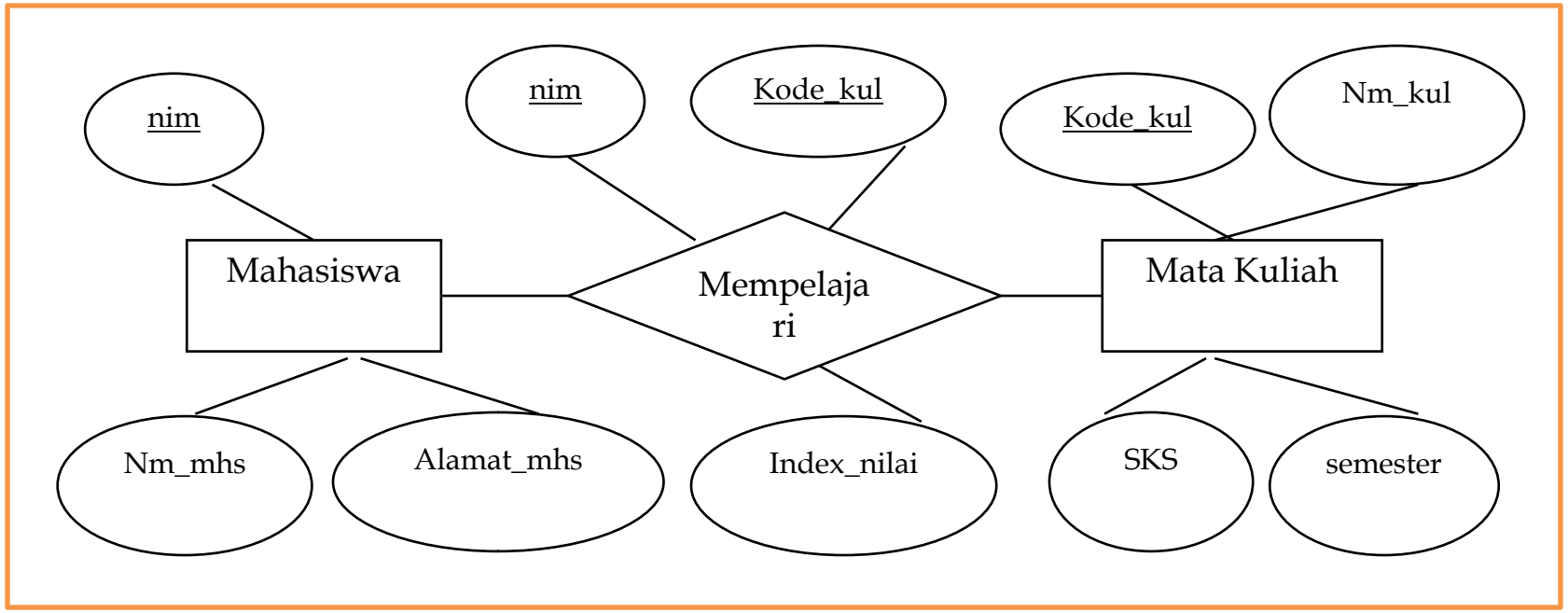
MODEL DATA

Model Data Keterhubungan antar Entitas (Entity Relationship data model)

- Menjelaskan hubungan antar data dalam sistem basis data berdasarkan suatu presepsi bahwa real world terdiri dari obyek-obyek dasar yang mempunyai hubungan relasi antara obyek-obyek tersebut
- Relasi antara obyek dilukiskan dengan menggunakan simbol-simbol grafis tertentu

MODEL DATA

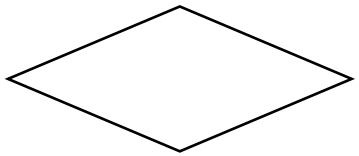
Contoh : Model Data Keterhubungan antar Entitas



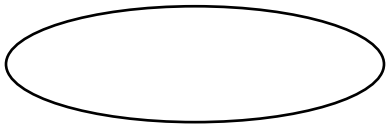
MODEL DATA

Keterangan simbol :

: menunjukkan obyek dasar/entitas (entity)



: menunjukkan relasi



: menunjukkan atribut dari obyek dasar/entitas

: menunjukkan adanya relasi/link

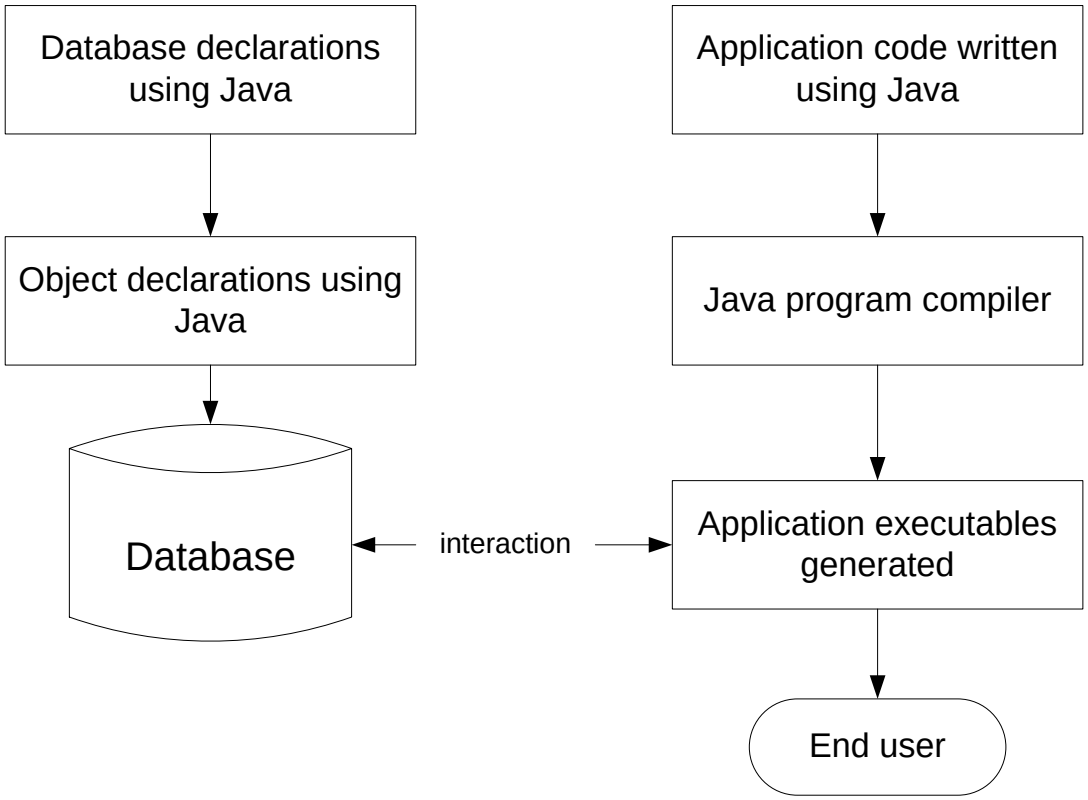
MODEL DATA

Model Data Berorientasi Obyek

- Model basis data berorientasi objek adalah suatu model basis data, dimana data didefinisikan, disimpan, dan diakses menggunakan pemrograman berorientasi objek.
- Basis data berorientasi objek didefinisikan dengan menggunakan bahasa pemrograman berorientasi objek, yaitu bahasa Java.
- Aplikasi End user juga di bangun dengan menggunakan bahasa berorientasi objek.
- Object database management system digunakan untuk membuat link antara basis data dan aplikasi.

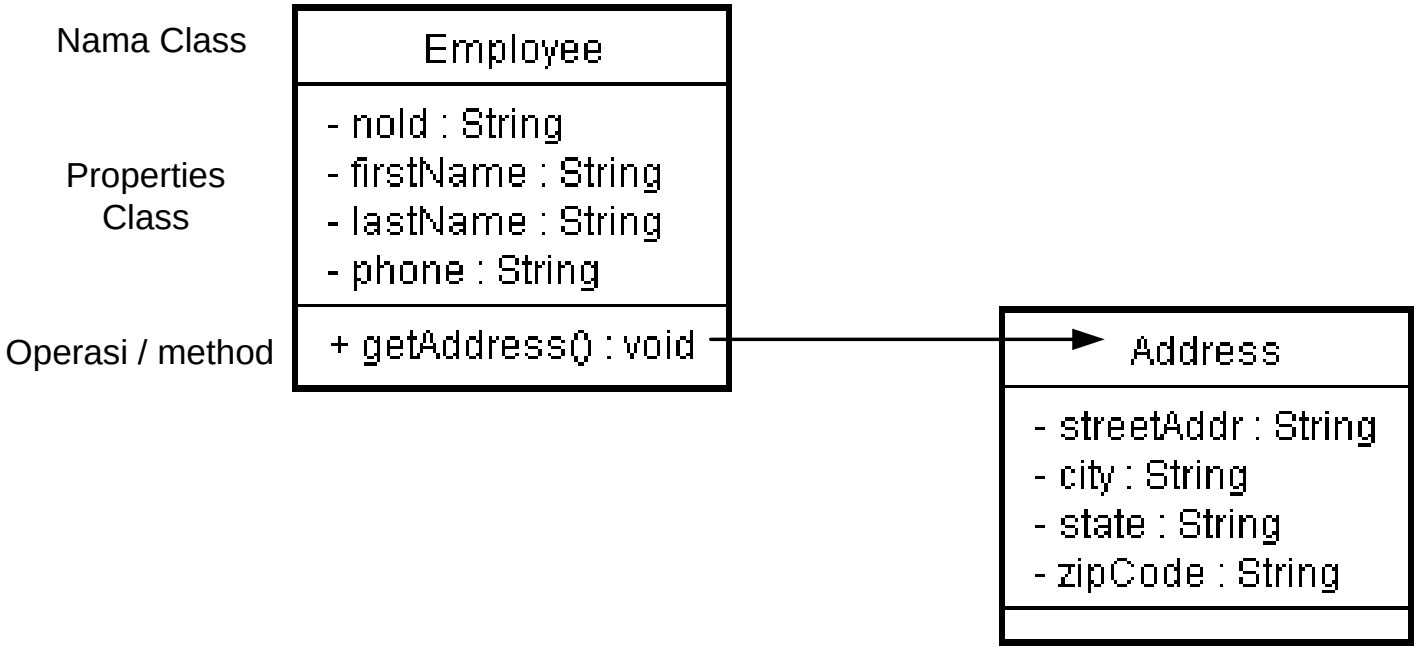
MODEL DATA

Contoh : Model Data Berorientasi Obyek



MODEL DATA

Relasi pada basis data berorientasi obyek



MODEL DATA

Kelebihan basis data berorientasi objek:

a. Programmer hanya dibutuhkan memahami konsep berorientasi objek untuk mengkombinasikan konsep berorientasi objek dengan storage basis data relasional
b. Objek dapat dilakukan sifat pewarisan dari objek yang lain
c. Secara teoritis mudah untuk mengatur objek
d. Model data berorientasi objek lebih kompatibel dengan tools pemrograman berorientasi objek.

MODEL DATA

Kelemahan basis data berorientasi objek:

User harus memahami konsep berorientasi objek, karena basis data berorientasi objek tidak dapat bekerja dengan metoda pemrograman tradisional

MODEL DATA

Latihan dan Soal

1. Sebelum beberapa vendor-vendor seperti Microsoft dan Oracle mengeluarkan DBMS, bagaimana orang atau perusahaan melakukan penyimpanan data. Jelaskan secara singkat !
2. Berikan alasan anda, mengapa model basis data flat-file sulit untuk dapat dilakukan relasi ?
3. Berikan perbedaan dan persamaan mengenai hubungan parent/child yang terdapat pada model basis data hirarki dan jaringan !
4. Bagaimana representasi model basis data relasional, berikan penjelasan secara singkat !

MODEL DATA

Latihan dan Soal

5. Untuk basis data yang terdiri dari beberapa tabel, bagaimana model basis data relasional dalam merelasikan tabel-tabel tersebut, berikan uraian secara singkat !
6. Mengapa model basis data relasional menjadi sangat populer, dan sejauh mana dukungan yang diberikan oleh vendor-vendor DBMS ?
7. Apa yang anda ketahui tentang model basis data berorientasi objek, berikan penjelasan secara singkat !
8. Bagaimana representasi model basis data berorientasi objek ?