**PENGANTAR DATABASE PERTEMUAN 07**

**PENGEMBANGAN PERANGKAT LUNAK**



|  |  |
| --- | --- |
| Abdullah Qa’id Mu’aadz | 22255601001 |

PeTIK II Jombang YBM PLN Tahun Akademik 2022/2023

Denanyar Utara, Plosogeneng,Kec. Jombang,

Kabupaten Jombang, Jawa Timur

Daftar Isi

[**PERTEMUAN 01 Pengantar Database** 5](#_Toc115896088)

[Definisi Data 5](#_Toc115896089)

[Jenis Data 5](#_Toc115896090)

[Fungsi Informasi 5](#_Toc115896091)

[Data vs Informasi 5](#_Toc115896092)

[Apa itu Database? 6](#_Toc115896093)

[DBMS ? 6](#_Toc115896094)

[Basis Data 6](#_Toc115896095)

[Sejarah Database 6](#_Toc115896096)

[Metadata 6](#_Toc115896097)

[Kelemahan Sistem Berbasis Data 7](#_Toc115896098)

[Permasalahan Dependensi Data 7](#_Toc115896099)

[Solusi Pendekatan dengan BASIS DATA 7](#_Toc115896100)

[Keuntungan Basis Data 7](#_Toc115896101)

[Komponen Basis Data 8](#_Toc115896102)

[**PERTEMUAN 02 Pemodelan Data** 9](#_Toc115896103)

[Data Model 9](#_Toc115896104)

[A. Object-based logical model 9](#_Toc115896105)

[**1.** **Entity Relationship Model** 9](#_Toc115896106)

[**2.** **Model Data Berbasis Objek** 10](#_Toc115896107)

[**3.** **Semantic Model** 11](#_Toc115896108)

[**4.** **Functional Data Model** 11](#_Toc115896109)

[B. Record-based logical model 11](#_Toc115896110)

[**1.** **Hirarchycal Model (Tree Structure)** 12](#_Toc115896111)

[**2.** **Network Model (Plex Structure)** 12](#_Toc115896112)

[**3.** **Relational Model** 12](#_Toc115896113)

[**PERTEMUAN 03 Model Relasional Database** 14](#_Toc115896114)

[Pemodelan Data Organisasi 14](#_Toc115896115)

[**Skema Basis Data** 14](#_Toc115896116)

[**Tingkatan Skema** 14](#_Toc115896117)

[**Contoh Skema Konsepsual** 15](#_Toc115896118)

[**Contoh Skema Logis** 15](#_Toc115896119)

[ERD ( Entity Relationship Diagram) 15](#_Toc115896120)

[**Notasi ERD (1)** 16](#_Toc115896121)

[**Notasi ERD (2)** 16](#_Toc115896122)

[**Model E-R : Entitas (1)** 16](#_Toc115896123)

[**Model E-R : Entitas (1)** 17](#_Toc115896124)

[Atribut 17](#_Toc115896125)

[**Komposit Atribut** 18](#_Toc115896126)

[**Atribut : Komposit , Multiple Value & Derived** 18](#_Toc115896127)

[**Notasi Relasi - ERD** 18](#_Toc115896128)

[**Relasi** 19](#_Toc115896129)

[**Relasi Antar Entitas** 19](#_Toc115896130)

[**PERTEMUAN 04 Normalisasi Database** 20](#_Toc115896131)

[Pendekatan Desain Database 20](#_Toc115896132)

[**Normalisasi** 20](#_Toc115896133)

[**Anomali** 21](#_Toc115896134)

[**Functional Dependency (FD)** 21](#_Toc115896135)

[**Kandidat Key** 22](#_Toc115896136)

[**PERTEMUAN 05 Pengantar SQL** 23](#_Toc115896137)

[Database ? 23](#_Toc115896138)

[**DBMS** 23](#_Toc115896139)

[**Database Relation** 23](#_Toc115896140)

[**Relational Model** 23](#_Toc115896141)

[**SQL – RDBMS** 24](#_Toc115896142)

[**Perintah SQL** 24](#_Toc115896143)

[MySQL 25](#_Toc115896144)

[**Software MySQL** 25](#_Toc115896145)

[**Fitur MySQL** 25](#_Toc115896146)

[Xampp Control Panel 26](#_Toc115896147)

[**Xampp Control Panel** 26](#_Toc115896148)

[**Laragon Control Panel** 26](#_Toc115896149)

[Login Ke Server MySQL 27](#_Toc115896150)

[**Create Table** 27](#_Toc115896151)

[**Jenis Field Table** 28](#_Toc115896152)

[**SQL-DDL :: Contoh CREATE TABLE** 28](#_Toc115896153)

[**SQL-DDL :: Contoh ALTER TABLE** 28](#_Toc115896154)

[**SQL-DDL :: Contoh INSERT** 29](#_Toc115896155)

[**PERTEMUAN 06 Perintah SQL Select 1** 30](#_Toc115896156)

[Perintah SELECT 30](#_Toc115896157)

[**SQL SELECT : Basic** 30](#_Toc115896158)

[**SQL SELECT : Where** 31](#_Toc115896159)

[**SQL SELECT : Update, Delete** 31](#_Toc115896160)

[**SQL SELECT : Order By** 32](#_Toc115896161)

[**Klausa DISTINCT** 32](#_Toc115896162)

[**SQL SELECT : DISTINCT** 32](#_Toc115896163)

[**Operator Perbandingan** 33](#_Toc115896164)

[**SQL SELECT : Between** 34](#_Toc115896165)

[**Keyword LIKE dan NOT LIKE** 34](#_Toc115896166)

[**Klausa LIMIT dan OFFSET** 35](#_Toc115896167)

[**PERTEMUAN 07 Perintah SQL Select 2 Built-in Function** 36](#_Toc115896168)

[SQL Data Type : Date & Time 36](#_Toc115896169)

[**Fungsi Waktu** 36](#_Toc115896170)

[**Fungsi String** 37](#_Toc115896171)

# **PERTEMUAN 01 Pengantar Database**

## **Definisi Data**

* **Navathe dan Elmasri, 2000:** Data yaitu fakta yang dapat disimpan dan memiliki arti
* **Hoffer, Prescott, dan McFadden, 2005:** Data yaitu sesuatu yang mewakilkan objek dan peristiwa yang memiliki arti dan sangat penting bagi pemakai atau user
* **Data:** fakta, teks, hasil pengukuran, gambar, suara, dan video yang bernilai informasi.

## **Jenis Data**

* **Structure data**
  + Tipe data yang dapat disimpan di database atau spreadsheet, diperlukan untuk dikelola sesuai dengan format penyimpanan standar dan ontologi, seperti : nama, alamat, telpon,
  + Contoh : Aplikasi sistem informasi akademik, aplikasi work flow, aplikasi SDM dll
  + Solusi Kelola: DBMS
* **Unstructured data**
  + text, audio, imagery, video
  + Contoh : data sistem email siswa, chat rooms, hasil questioner, video / audio di sistem e-elearning , RFID , barcode
  + Solusi Kelola: BIGDATA (5V: Volume, Velocity, Varieaty, Veracity, Value)

Informasi - dapat dimanfaatkan sebagai dasar untuk pengambilan keputusan dan memahami permasalahan/situasi

## **Fungsi Informasi**

I = I (D, S, T)

* I : Informasi
* D : Data
* S : Pengetahuan
* T : Waktu

## **Data vs Informasi**

* **Data** : fakta mentah, baik berupa kumpulan angka, kata-kata, simbol, ataupun sebuah citra yang didapat dari hasil observasi atau referensi sumber yang dapat diolah menjadi bentuk lain
* **Informasi** : data yang berbentuk angka, kata-kata, simbol, ataupun citra yang telah diolah, diproses, serta dikelola dengan baik sehingga data mentah itu dapat menjadi sesuatu yang dimengerti serta bermanfaat bagi penerimanya

## **Apa itu Database?**

* Sebuah database adalah terpusat dan kumpulan yang terstruktur dari data yang tersimpan dalam sebuah sistem computer
* Database memiliki kemampuan untuk : pengambilan (retrieving), penambahan (adding), pemodifikasian (modifying), dan penghapusan (deleting) dari data saat dibutuhkan
* Database memiliki kemampuan untuk melakukan proses transformasi pengambilan data menjadi informasi yang berguna.
* Sebuah database biasanya dikelola oleh seorang: Database Administrator (DBA).

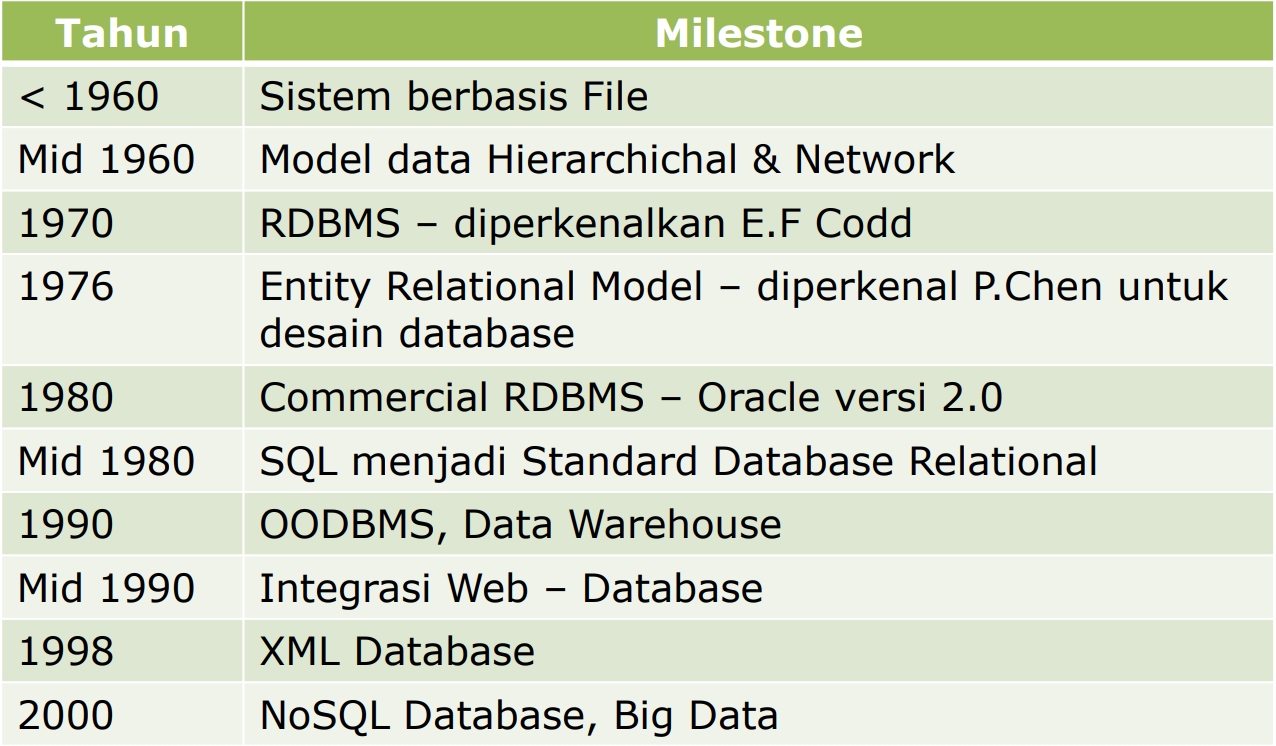
## **DBMS ?**

* Database Management System (DBMS) adalah sistem penyimpanan dan pengambilan data yang memungkinkan data untuk disimpan secara **tidak terduplikasi** dan mengorganisasikan data dengan **struktur** yang dikehendaki penggunanya.

## **Basis Data**

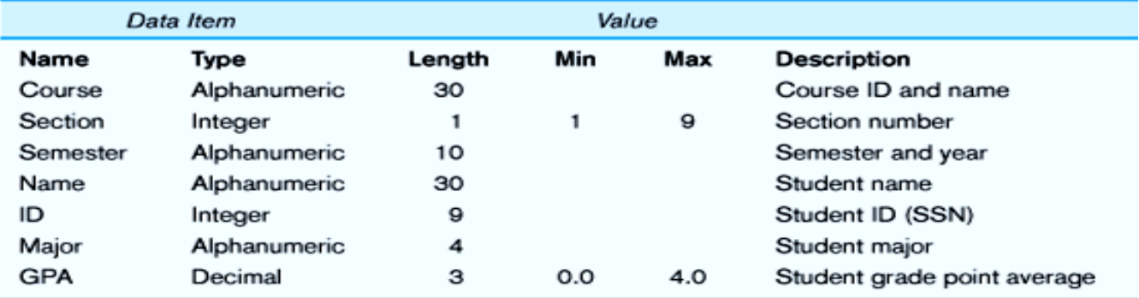
* **Basis Data (Database):** kumpulan data yang terorganisir berdasarkan suatu struktur hubungan (konsep entitas).

## **Sejarah Database**

****

## **Metadata**

Deskripsi tentang format dan karakteristik data, termasuk tipenya, ukurannya, nilai-nilai yang absah, dan dokumentasi lainnya.



## **Kelemahan Sistem Berbasis Data**

* Dependensi antara program & data
  + Fungsi penyimpanan dan akses data merupakan bagian tak terpisahkan dari setiap program aplikasi.
* Redundansi (duplikasi) data
  + Masing-masing aplikasi/program memiliki kopi sendiri untuk data yang sama.
* Inkonsistensi data
  + Tidak ada pengawasan data secara terpusat.
* Sulit berbagi (sharing) data
  + Format file antar program aplikasi dapat berbeda-beda.
* Biaya pemeliharaan tinggi

## **Permasalahan Dependensi Data**

* Setiap program aplikasi harus memiliki data sendiri – memungkinkan duplikasi data.
* Setiap program aplikasi harus berurusan dengan metadata (format data) untuk setiap file yang digunakan.
* Setiap program aplikasi harus memiliki fungsi untuk membaca, menulis, mengubah dan menghapus data.
* Tidak ada koordinasi pengelolaan data yang sama antar aplikasi.
* Sulit membakukan format-format file.

## **Solusi Pendekatan dengan BASIS DATA**

***Database***

* Pusat repositori data bagi seluruh organisasi.
* Data dikelola oleh suatu sistem pengendali.
* Data disimpan dalam format yang baku dan mudah dimengerti.

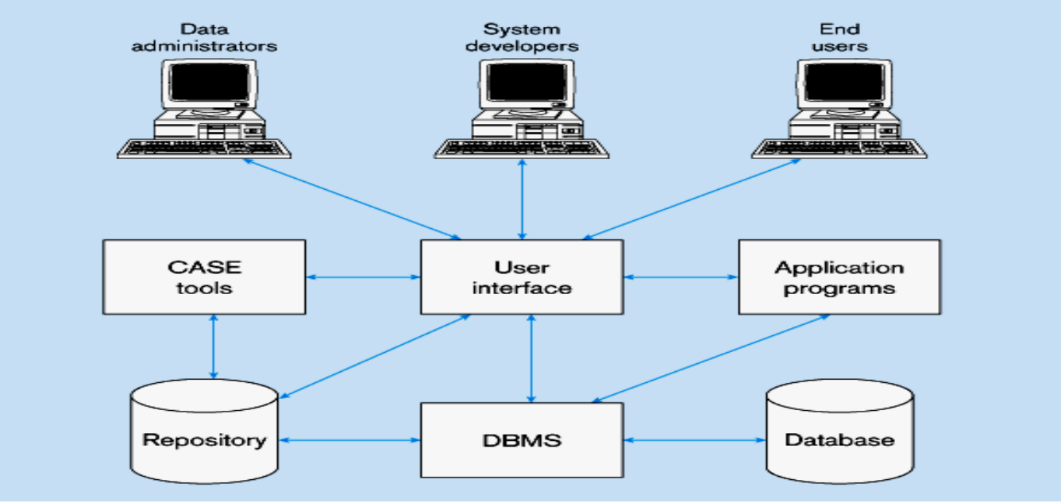
Membutuhkan suatu *Database Management System* (DBMS).

## **Keuntungan Basis Data**

* Independensi Program-Data
  + **Metadata disimpan dalam DBMS, sehingga aplikasi-aplikasi tidak harus berurusan dengan masalah format data.**
  + **Akses dan operasi pada data dikelola oleh DBMS, sehingga aplikasi tidak harus menjalankan prosedur akses data sendiri.**
  + Hasilnya: peningkatan produktivitas pengembangan dan pemeliharaan program aplikasi.
* Duplikasi Data Minimal
  + Meningkatkan integritas/konsistensi data.
* Memungkinkan berbagi data
  + Dengan perspektif (view) atas data yang mungkin berbeda antar aplikasi/pemakai.
* Memungkinkan pemberlakuan standar
  + Semua akses ke data dilakukan melalui jalur dan cara yang sama.
* Meningkatkan mutu data
  + Dengan memberlakukan konstrain dan aturan-aturan validasi data.
* Meningkatkan kemudahan implementasi akses data
  + Menggunakan bahasa query data standar (SQL, dsb.)
* Adanya mekanise pengamanan serta koordinasi transaksi dan akses secara bersamaan (concurrency)
  + Menjaga integritas data.

## **Komponen Basis Data**

* **Repositori** – pusat penyimpanan metadata.
* **Database** **Management System (DBMS)** – perangkat lunak untuk mengelola database.
* **Database** – pusat penyimpanan data.
* **Program Aplikasi** – perangkat lunak pengguna data.
* **User Interface** – fasilitas interaksi antara pengguna dan data secara tekstual atau grafis.
* **CASE Tools** – computer-aided software engineering



# **PERTEMUAN 02 Pemodelan Data**

## **Data Model**

* **Data Model** adalah sekumpulan konsep yang digunakan untuk menjelaskan struktur logika dari basis data (database structure) dan memberikan gambaran tingkat-tingkat abstraksi data (data abstraction).
* Data model juga mencakup sekumpulan operasi (basic & user-defined) yang dapat dilakukan terhadap data yang dihimpun dalam basis data.
* Operasi dasar (basic) disediakan oleh sistem basis data: **insert, delete, update, dan retrieve.**
* **User-defined operation** dibuat oleh perancang, contoh: operasi menghitung IPK mahasiswa
* **Data abstraction** adalah deskripsi mengenai struktur basis data yang mudah dimengerti oleh user, atau bisa juga mencakup detil dari penyimpanan data yang biasanya perlu diketahui oleh perancang basis data jadi ada tingkatan abstraksi.
* **Database structure** mencakup data type (type of name is string), relationship (customer has relationship with account), dan constraint (contoh: required data, max/min data, default data)

**Data model dapat dibagi menjadi 2 kelompok besar:**

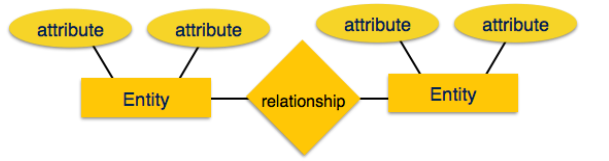
1. **Object-based logical model:**
   1. Entity-relationship model
   2. Object-oriented model
   3. Semantic model
   4. Fungsional model
2. **Record-based logical model:**
   1. Relationship model
   2. Network model
   3. Hierarchical model

## **A. Object-based logical model**

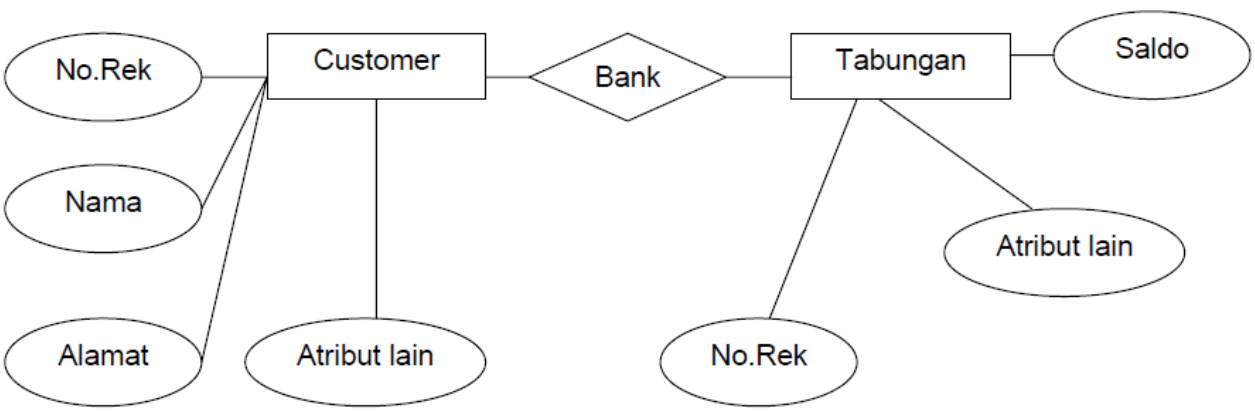
### **Entity Relationship Model**

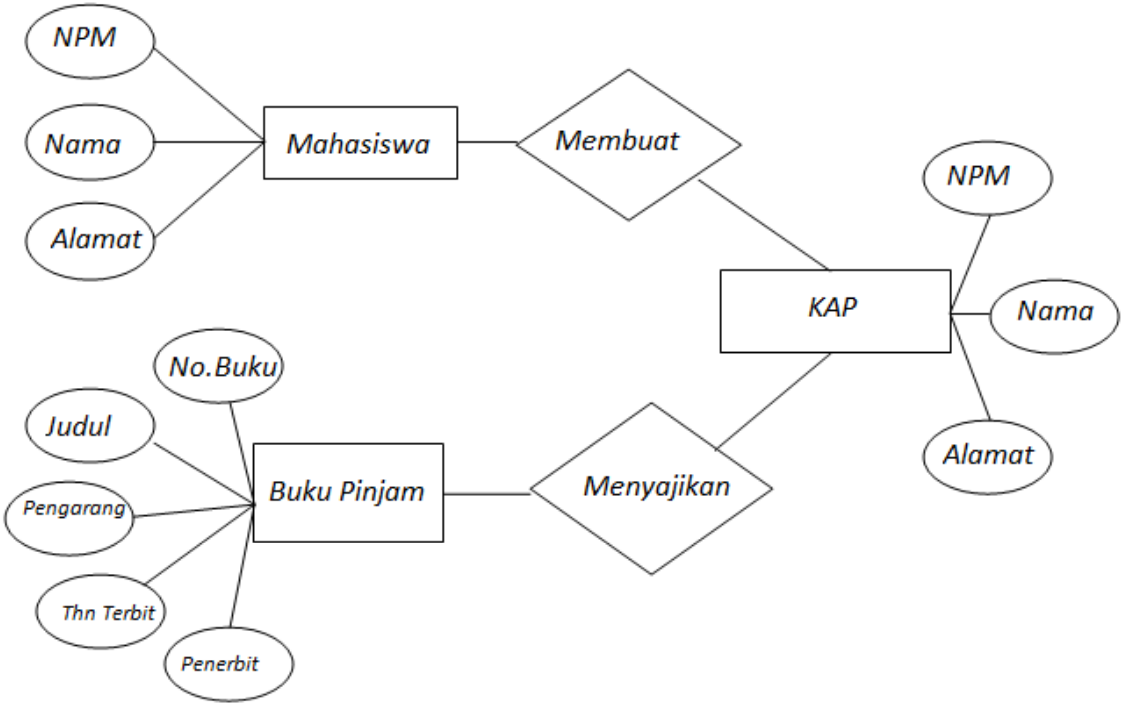
Merupakan himpunan data dan relasi yang menjelaskan hubungan logik antar data dalam suatu basis data berdasarkan objek datanya.

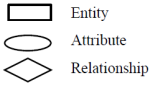
Merupakan model untuk menjelaskan hubungan antar data dalam basis data berdasarkan persepsi bahwa dunia nyata terdiri dari objek-objek dasar yang mempunyai relasi diantara objek tersebut.



**Contoh Entity Relationship Model**

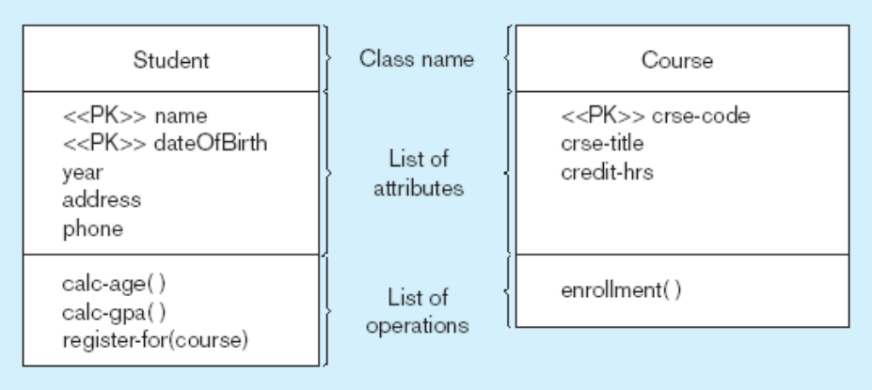






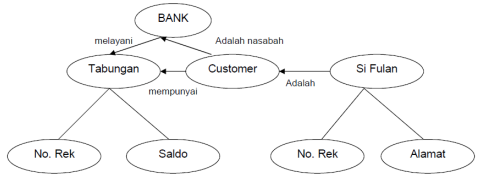
### **Model Data Berbasis Objek**

* Data di representasikan dalam bentuk object di dunia nyata
* Class : blue print dari object
* Object instan dari class



### **Semantic Model**

Relasi antar objek dinyatakan dengan kata-kata (semantic)



Arti tanda :

: Menunjukkan adanya relasi

: Menunjukkan atribut

### **Functional Data Model**

* Functional model : menggambarkan aliran data dari inputan eksternal, melalui proses operasi-operasi) dan penyimpanan internal
* Digambarkan sebagai diagram aliran data (Data Flow Diagram)

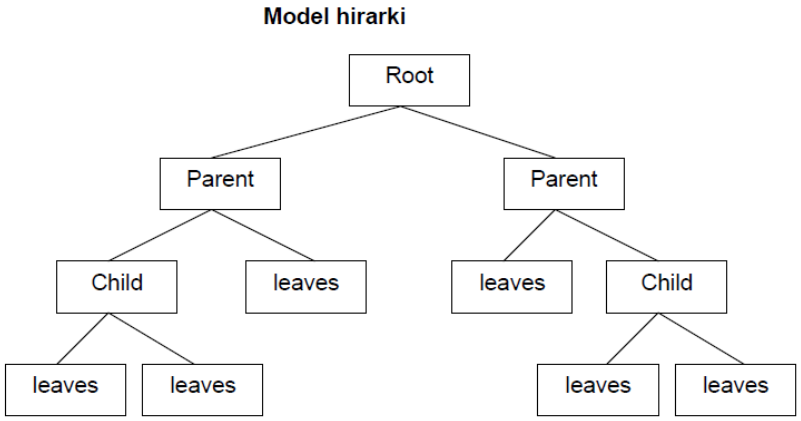
|  |  |
| --- | --- |
|  |  |
|  |  |

## **B. Record-based logical model**

* Implementasi dari konsep data model yang dapat dimengerti oleh end user, menggambarkan organisasi data dalam komputer, tanpa detil penyimpanan dalam komputer.
* Disebut sebagai record-based data model, karena merepresentasi data dalam bentuk record structure.
* Konsep ini digunakan untuk menjelaskan skema tradisional database seperti hierarchical databases, network model dan relational database.

### **Hirarchycal Model (Tree Structure)**

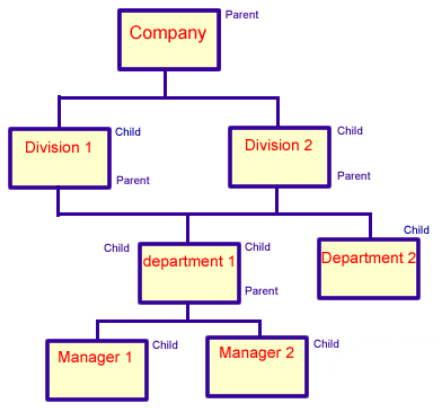
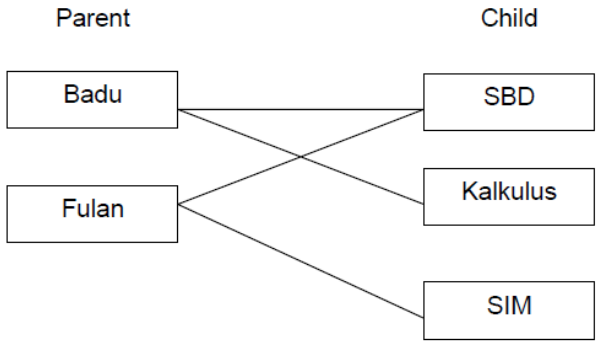
Menjelaskan tentang hubungan logik antar data dalam bentuk hubungan bertingkat (hirarki). Dimana elemen penyusunnya disebut node, yang berupa rinci data, agregat data, atau record.



### **Network Model (Plex Structure)**

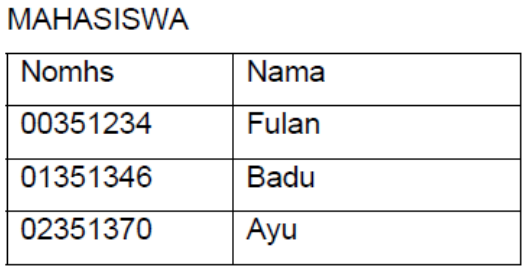
Hampir sama dengan hirarki model, namun sebuah child pasti berada pada lebih rendah dari pada parent, dan sebuah child dapat mempunyai beberapa parent.

Contoh :

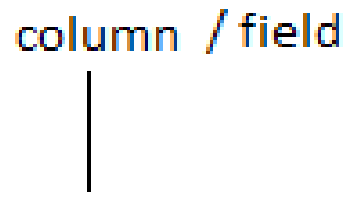


### **Relational Model**

Menjelaskan hubungan logik antar data dalam basis data dengan menvisualkan ke dalam tabel-tabel yang terdiri dari sejumlah baris dan kolom yang menunjukan atribut tertentu.

Keterangan :

* Jumlah kolom disebut egree, ada 2
* Kolom disebut juga atribut (field), ada 2
* Tiap baris disebut record / tuple, ada 3 record
* Banyaknya baris dalam satu table disebut *cardinality*

 **Tabel**

* **Field/Column** – Satu jenis informasi/data yang Mempunyai Tipe Data Sama
* **Record/Row** – Satu kesatuan informasi yang terdiri atas satu Field atau lebih
* **Character** – Satuan terkecil dari data

Contoh Relational Model :



**Keuntungan Relational Model**

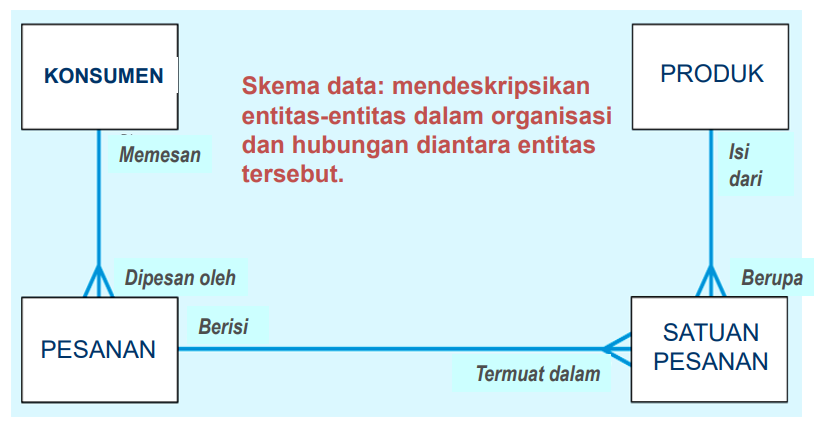
Simple & Elegan

* Database adalah kumpulan dari satu atau lebih dari relasi, dimana setiap relasi adalah berupa tabel, kolom dan baris
* Tampilan data berbentuk tabular mudah dimengerti
* Kemudahan tampilan data walaupun dengan perintah query yang rumit

# **PERTEMUAN 03 Model Relasional Database**

## **Pemodelan Data Organisasi**

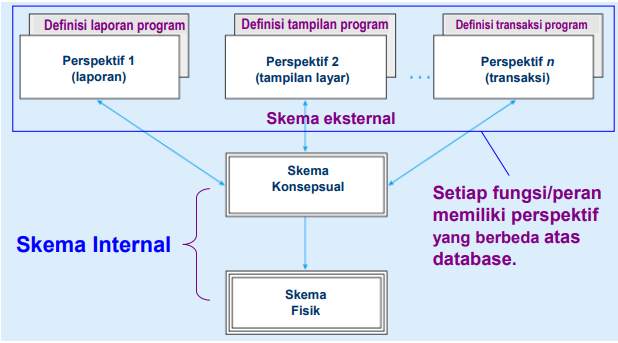
* Adalah langkah pertama dalam pengembangan database.
* Mendefinisikan batasan ruang lingkup informasi.
* Memberikan gambaran umum tentang data organisasi.
  + Berupa diagram Entity Relationship (ER).
  + Skema database, mendeskripsikan:
    - Tipe-tipe entitas data.
    - Hubungan antar entitas.
    - Peraturan/ketentuan bisnis organisasi.

Contoh :

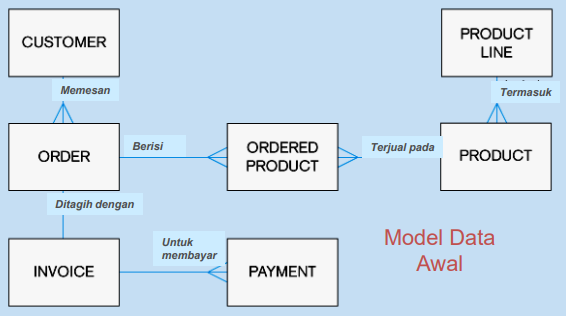
### **Skema Basis Data**

* Skema Logis
  + Struktur data dalam database sesuai dengan teknologi DBMS yang dipilih.
* Skema Fisik
* Struktur fisik penyimpanan: tipe dan format data, struktur lokasi penyimpanan, dsb.

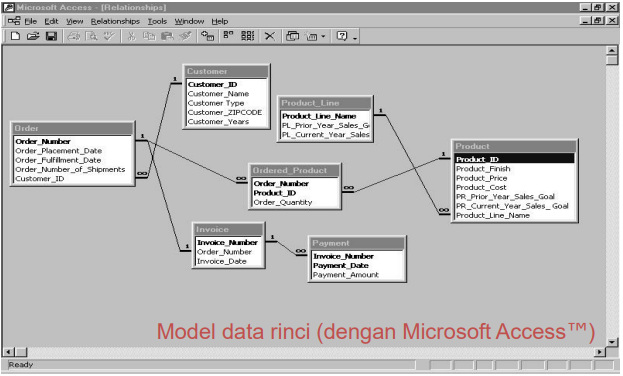
### **Tingkatan Skema**



### **Contoh Skema Konsepsual**



### **Contoh Skema Logis**

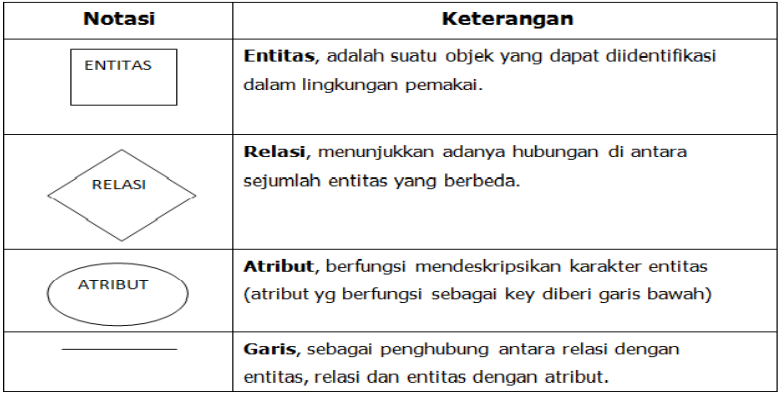


## **ERD ( Entity Relationship Diagram)**

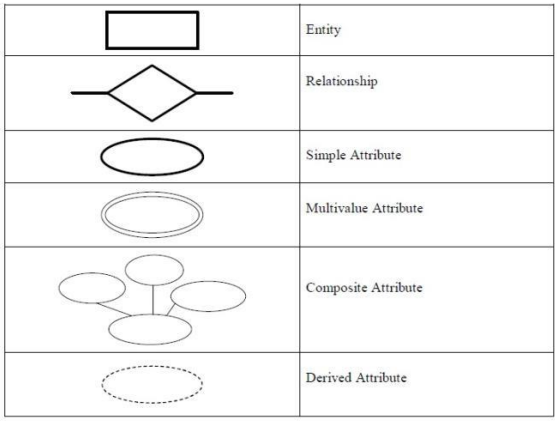
Suatu grafik khusus yang menggambarkan keterkaitan antara entitas dalam database. Dalam diagram ER terdapat simbol-simbol untuk mewakili tiga jenis informasi yang berbeda.

Box digunakan untuk mewakili **entitas**. **Berlian** digunakan untuk mewakili hubungan dan **oval** digunakan untuk mewakili atribut

### **Notasi ERD (1)**

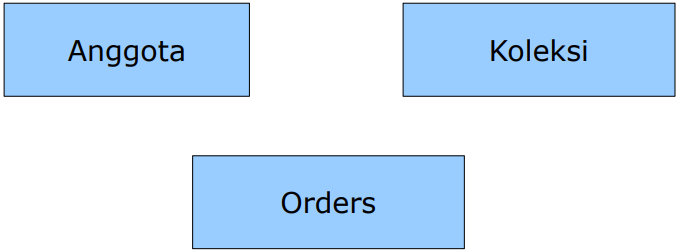


### **Notasi ERD (2)**



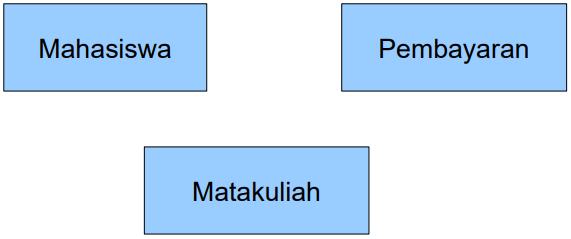
### **Model E-R : Entitas (1)**

* Object atau Konsep yang terlibat dalam suatu bisnis proses



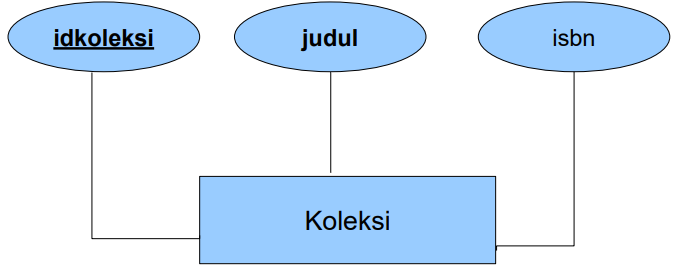
### **Model E-R : Entitas (1)**

* Object atau Konsep yang terlibat dalam suatu bisnis proses

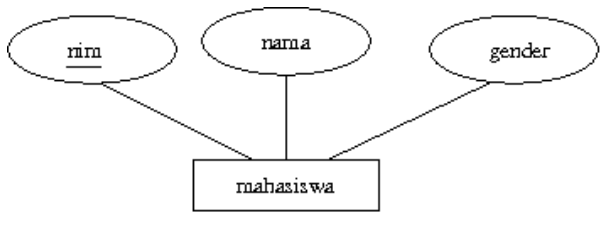
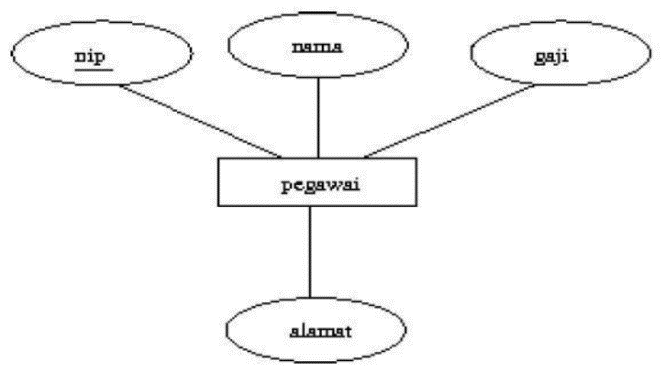


## **Atribut**

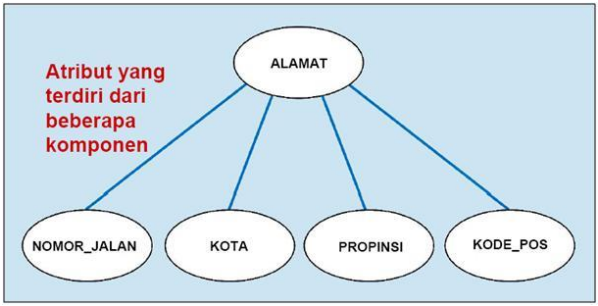
* Properti atau item pada suat entitas



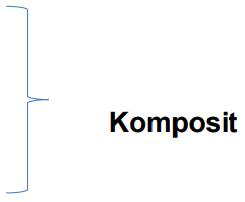
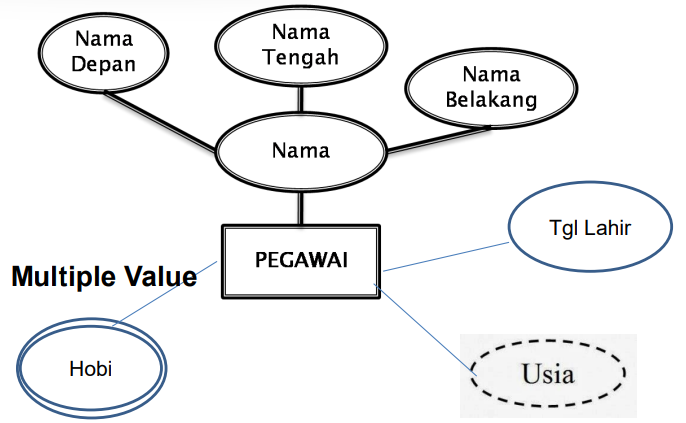
* Properti atau item pada suatu entitas

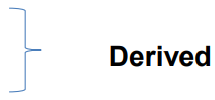


### **Komposit Atribut**

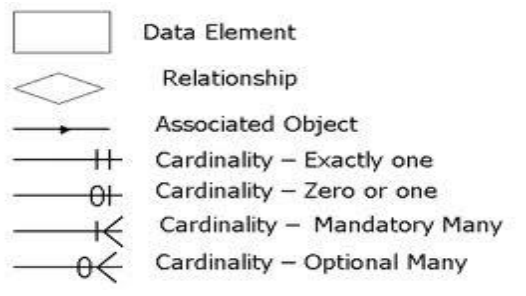


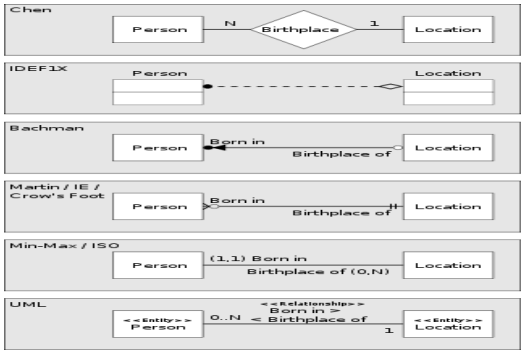
### **Atribut : Komposit , Multiple Value & Derived**





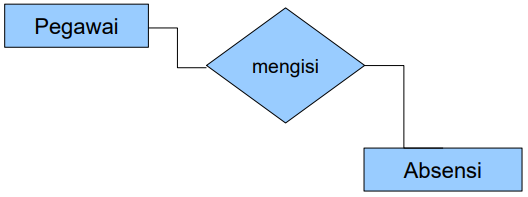
### **Notasi Relasi - ERD**





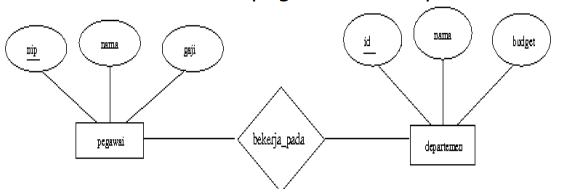
### **Relasi**

* Hubungan yang terjadi antara entitas

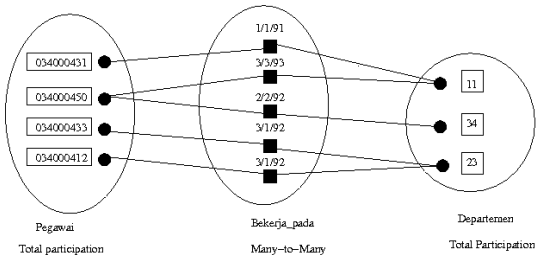


### **Relasi Antar Entitas**

* Relasi antara entitas pegawai dan departemen



* Himpunan relasi pegawai dan departemen



# **PERTEMUAN 04 Normalisasi Database**

## **Pendekatan Desain Database**

Top Down :

* Pendekatan desain database dengan disesuaikan dengan requirement / bisnis proses organisasi
* Sumber data: SOP , Aturan Bisnis (role business), Form2, wawancara.
* Output akhir : Desain ERD

Bottom Up :

* Pendekatan desain database dengan bersumber pada data-data yang sudah ada (normalisasi data)
* Sumber data: Laporan2, spreadsheet
* Output akhir : Desain ERD

### **Normalisasi**

Proses yang dilakukan pada tahap perancangan logis yang bertujuan menormalisasi tabel-tabel dalam database relasional.

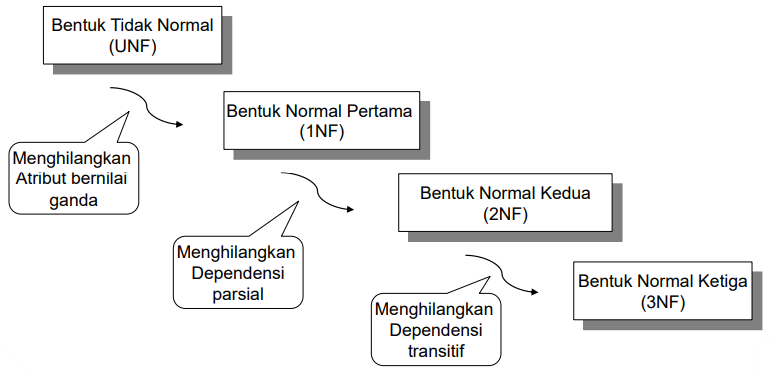
**Normalisasi skema logis**

* Melakukan validasi dan penyempurnaan rancangan logis database sehingga memenuhi format tertentu (*form normal*) untuk **meminimasi duplikasi** **data**.
* Definisi teknis: proses dekomposisi atau memecah relasi-relasi yang mengandung anomali menjadi relasi-relasi kecil yang **terstruktur dengan baik** (*memenuhi* *form normal*).

**Alasan Normalisasi Database**

* Meminimalisasi data berulang (redundansi data)
* Proses dilakukan minimal data berulang tidak muncul pada bentuk normal ke 3 (3NF)
* Menghilangkan anomali-anomali
  + Anomali penambahan ( INSERT )
  + Anomali penghapusan ( DELETE )
  + Anomali pengubahan (UPDATE)

#### **Langkah-Langkah Normalisasi**



### **Anomali**

Anomali adalah : masalah yang timbul dalam relasi Ketika terjadi operasi pemutahiran data dalam relasi

1. **Anomali Penambahan Data**, Penambahan record baru mengharuskan duplikasi nilai-nilai atribut tertentu.
2. **Anomali Penghapusan Data**, Penghapusan record dapat mengakibatkan hilangnya nilai atribut yang hanya terdapat pada record tsb. (yang mungkin masih dibutuhkan).
3. **Anomali Pengubahan Data**,Pengubahan nilai atribut suatu record mengharuskan pengubahan atribut yang sama pada banyak recordrecord lain.

#### **Penyebab Anomali**

Mengapa anomali-anomali ini terjadi?

Karena menggabungkan dua tema (konsep entitas) dalam satu relasi. Ini mengakibatkan duplikasi-duplikasi sebagai akibat dari ketergantungan antar atribut yang tidak pada tempatnya.

### **Functional Dependency (FD)**

Tahap pertama normalisasi setelah terbentuk 1NF, Tentukan terlebih dahulu **Functional Dependency (FD)** atau **Ketergantungan Fungsional**, khususnya dalam melakukan dekomposisi rancangan database.

Simbol Fungtional Dependency (FD) :

A B : B memiliki ketergantungan dengan A

### **Kandidat Key**

Dependensi Fungsional

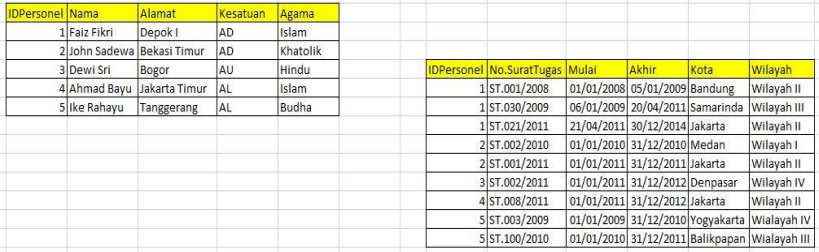
Nilai suatu atribut yang disebut ***atribut determinan*** dalam suatu relasi menentukan nilai dari atribut yang lain.

* Kandidat Key adalah atribut determinan
  + Atribut pengenal dengan nilai unik.
  + Salah satu dari kandidat-kandidat key dipilih menjadi primary key.
  + Setiap atribut yang bukan primary key (non-key) secara fungsional tergantung (dependen) pada primary key.

**Contoh : Functional Dependency**



**Bentuk 2NF**



**Bentuk 3NF : Normal Form Ketiga**

Bentuk 2NF plus tidak ada dependensi transitif: Satu atribut secara fungsional menentukan nilai atribut kedua yang pada gilirannya menentukan nilai atribut ketiga



Perhatikan atribut kota dan wilayah

Kota bergantung pada wilayah ::

IDPersonel, No.SuratTugas → Mulai, Akhir, Kota, Wilayah

Wilayah → Kota

# **PERTEMUAN 05 Pengantar SQL**

## **Database ?**

Database Management System (DBMS) adalah sistem

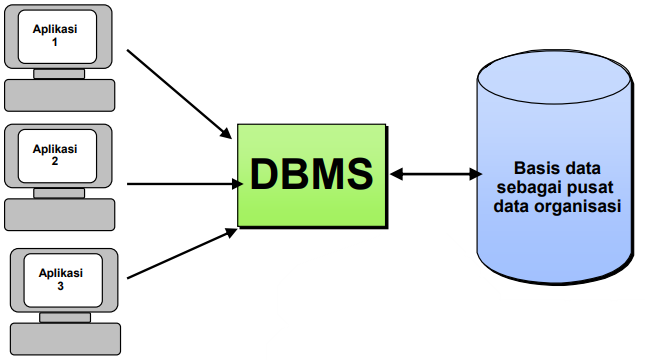
penyimpanan dan pengambilan data yang memungkinkan

data untuk disimpan secara tidak terduplikasi dan

mengorganisasikan data dengan struktur yang dikehendaki

penggunanya.

### **DBMS**



DBMS mengelola sumber daya data sebagaimana operating system (OS) mengelola sumber daya perangkat keras

### **Database Relation**

**EF Codd (1970):**

Memperkenalkan model relasional data, pada saat itu sebagian besar sistem database berdasarkan dua model data: Model hirarki (hierarchical model) dan model jaringan (network model). Prototipe sistem database model relasional dikembangkan di IBM dan di UC-Berkeley pada pertengahan tahun 1974

### **Relational Model**

**Simple & Elegan**

* Database adalah kumpulan dari satu atau lebih dari relasi, dimana setiap relasi adalah berupa tabel, kolom dan baris

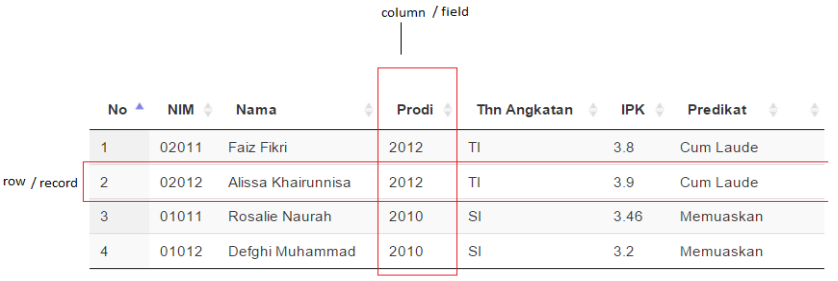
**Keuntungan**

* Tampilan data berbentuk tabular mudah dimengerti Kemudahan tampilan data walaupun dengan query yang rumit

- Field/Column – Satu jenis informasi/data yang Mempunyai Tipe Data Sama

- Record/Row – Satu kesatuan informasi yang terdiri atas satu Field atau lebih

- Character – Satuan terkecil dari data



### **SQL – RDBMS**

* SQL (Structure Query Language) - adalah suatu bahasa query yang dikembangkan oleh IBM pada projek DBMS relasional system - R.
* Selama beberapa tahun SQL menjadi bahasa query yang digunakan untuk melakukan pendefinisian skema, manipulasi data dan query pada DBMS relasional.

**Standarisasi SQL**

* Banyak digunakan vendor pada produk RDBMS
* Dibutuhkan standar penulisan SQL.
* Produk RDBMS diharapkan mengacu pada suatu standar yang ditetapkan.
* Diharapkan tidak ada perbedaan perintah SQL pada produk yang dikeluarkan.
* American National Standards Institute (ANSI) mengeluarkan Standard SQL
  + 1986: SQL-86
  + 1989: SQL-89 ; perubahan minor
  + 1992: SQL-92 ; ANSI+ISO – perubahan besar
  + 1999: SQL-99
  + 2003: SQL-2003

### **Perintah SQL**

* **Data Definition Language (DDL**) – Perintah SQL untuk mendefinisikan skema database atau table : CREATE, DROP, ALTER
* **Data Manipulation Language (DML)** – Perintah SQL untuk manipulasi data dan pencarian data : INSERT,UPDATE,DELETE,SELECT
* **Data Control Language (DCL)** – Perintah SQL untuk mengontrol database atau table : GRANT, REVOKE

## **MySQL**

* MySQL : RDBMS Server ( Client & Server )
* Dikembangkan oleh Michael Widenius & David Axmark 1994
* Lisensi GNU GPL / Proprietary dengan EULA
* Sekarang dimiliki oleh Sun Microsystems , dibawah Oracle Corporation
* Dibuat dengan kode C dan C++
* Berjalan di berbagai platform Sistem Operasi ( Windows, Linux, Mac,Solaris)
* Disupport beberapa Engine Database ( MyISAM, InnoDB)
* Disupport berbagai bahasa pemrograman ( tersedianya API / library untuk akses MySQL)
* Versi terakhir MySQL 5.7 ( [www.mysql.com](http://www.mysql.com))
* MySQL - MariaDB : database Open Source Based On MySQL

### **Software MySQL**

* Pada Sistem Operasi Linux telah menjadi satu paket dengan distribusi linux ( distro)
* Pada Sistem Operasi Windows bisa didownload di [www.mysql.com](http://www.mysql.com)
* Paket aplikasi LAMP / WAMP ( Linux / Windows Apache MySQL PHP )
  + XAMPP ( apachefreind.org )
  + WAMP
  + Laragon ( laragon.org)
* Paket XAMPP / Laragon : MariaDB
* Linux Ubuntu : MariaDB

### **Fitur MySQL**

* Standard ANSI
* Support Cross-platform
* Stored Procedures, Triggers , View
* Information schema
* SSL (Secure Socket Layer)
* Independent Storage Engine ( MyISAM, InnoDB, etc)
* Transaction ( InnoDB Engine)
* SUB SELECT
* Referential Integrity ( InnoDB Engine)
* Full Text Indexing (MyISAM Engine)
* Replication ( Master-Master & Master-Slave)
* Hot backup ( copy )

## **Xampp Control Panel**

* Aktifkan Pada sistem Linux
  + # service mysqld start

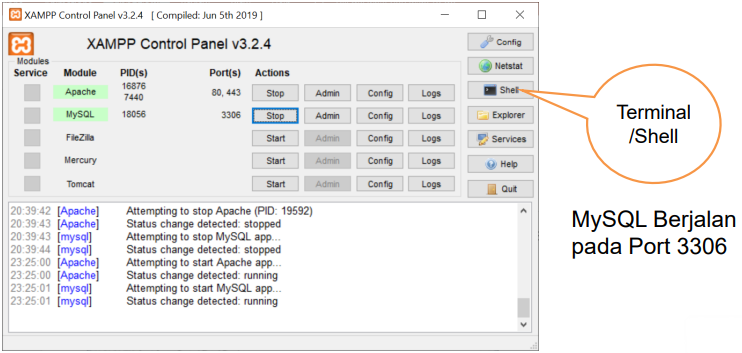
Atau

* + # /etc/init.d/mysqld start
* Matikan server MySQL pada sistem Linux
  + #service mysqld stop

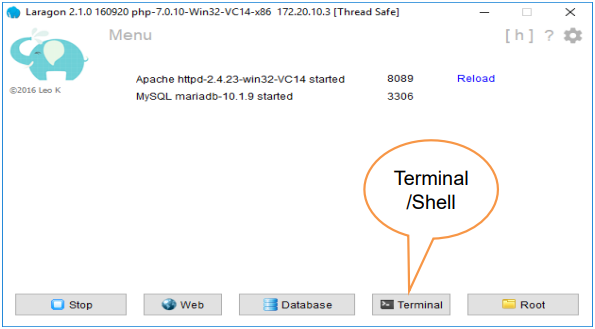
Atau

* + #/etc/init.d/mysqld stop
* Pada sistem Windows / XAMPP
  + Gunakan Control Panel XAMPP

### **Xampp Control Panel**



### **Laragon Control Panel**



## **Login Ke Server MySQL**

**Aktifkan Pada sistem Linux**

* Buka konsole /terminal ketikan

$ mysql

Atau

$ mysql -u root -p

* Selanjutnya anda berada dalam prompt mysql
* User root adalah user admin mysql

**Pada sistem Windows**

* Buka command prompt, dan masuk ke direktory instalasi mysql

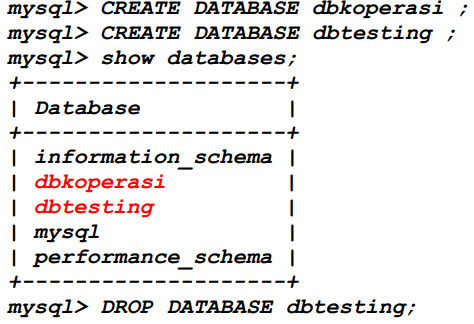
c:\> cd xampp\mysql\bin

* Jalankan perintah

C:\xampp\mysql\bin> mysql – u root -p

* Selanjutnya anda berada dalam prompt mysql

**Create Database**



### **Create Table**

Buat TABLE

* CREATE TABLE nama\_table (

col\_name column\_definition [ Constraint ] ,

col\_name column\_definition [ Constraint ] ,

...

) ;

* col\_name : nama field / kolom
* column\_defefinition : tipe data
* constraint: opsional
  + Key Field : PRIMARY KEY, UNIQUE KEY
  + Not NULL : Wajib Diisi
  + DEFAULT nilai\_default : nilai default data ketika di INSERT

### **Jenis Field Table**

Primary Key

* Field kunci dari suatu table
* Data bersifat unik ( tidak boleh ada yang sama)
* Di index oleh DBMS

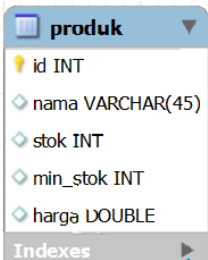
Foreign Key

* Field yang mengacu ke table lain
* Berperan sebagai duta dari table lain (table acuan)
* Detail data berada pada table acuan
* Biasanya adalah primary key di table acuan

Unique Key

* Data bersifat unik ( tidak boleh ada yang sama)
* Secara default tidak di index oleh DBMS

### **SQL-DDL :: Contoh CREATE TABLE**

Buat TABLE

* CREATE TABLE produk (

id integer auto\_increment primary key,

nama varchar(45) not null,

stok integer default 0,

min\_stok integer default 0,

harga double default 0

) ;

### **SQL-DDL :: Contoh ALTER TABLE**

Mengubah nama table

* ALTER TABLE nama\_table lama RENAME nama\_tablebaru
* Mengubah nama table produk menjadi table barang
  + ALTER TABLE produk RENAME barang ;

Mengubah nama field / tipe data

* ALTER TABLE nama\_table CHANGE nama\_fieldlama nama\_field baru tipe\_data
* Mengubah field nama menjadi nama\_produk dengan tipe data varchar(30)
  + ALTER TABLE produk CHANGE nama nama\_produk VARCHAR(30)

Menambah field baru

* ALTER TABLE nama\_table ADD nama\_fieldbaru tipedata [ AFTER nama\_field ]
* Menambah field jenis pada table produk ( akan diletakan pada urutan terakhir )
  + ALTER TABLE produk ADD jenis VARCHAR(30) ;
* Menambah field kode setelah field id
  + ALTER TABLE produk ADD kode VARCHAR(10) AFTER id ;

Menghapus field

* ALTER TABLE nama\_table DROP nama\_field\_yang\_dihapus
  + ALTER TABLE produk DROP harga ;

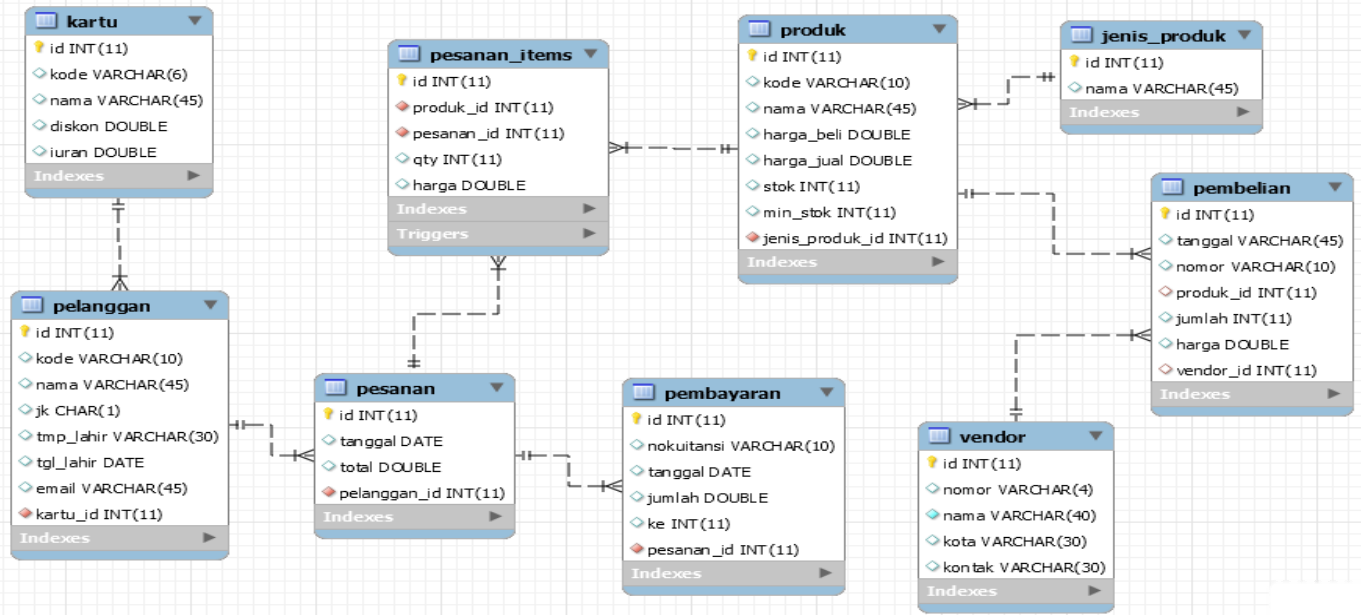
### **SQL-DDL :: Contoh INSERT**

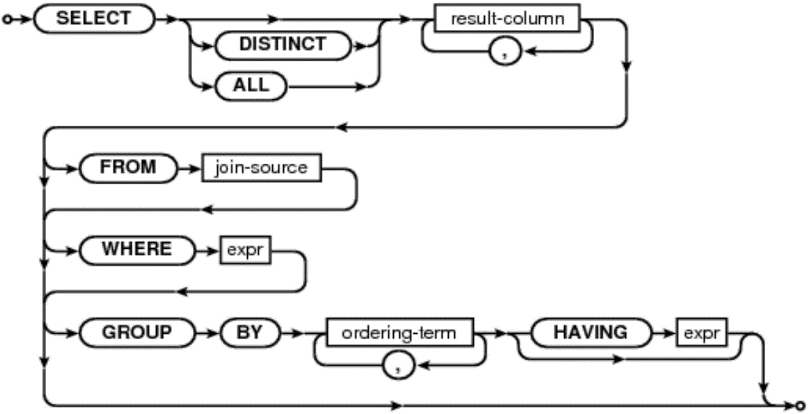
INSERT INTO nama\_table [ nama\_field ... ] VALUES (data ... )

* Menambah data ke table produk : hanya field id,nama dan stok
  + INSERT INTO produk (id,nama,stok) VALUES (DEFAULT, ‘Mie Instan’ , 200 );
* Menambah data ke table produk seluruh field
  + INSERT INTO produk VALUES (default,’Gula’, 200, 10, 8500);
* Menambah 2 data sekaligus ke table produk
  + INSERT INTO produk VALUES (default,’Kopi’,100,10,6500) , (default,’Teh’,30,5,2000);

# **PERTEMUAN 06 Perintah SQL Select 1**

ERD Database dbkoperasi





## **Perintah SELECT**

*Urutan Perintah :*

- SELECT

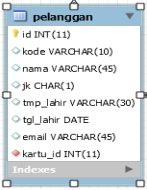
- FROM

- WHERE

- GROUP BY

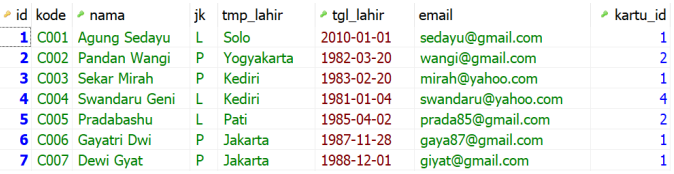
- HAVING

### **SQL SELECT : Basic**

- Problem: Menampilkan seluruh data pelanggan



- Result: 14 records



- Problem: Menampilkan kode, nama dan tmp\_lahir seluruh data pelanggan



- Result: 14 records

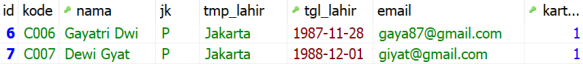


### **SQL SELECT : Where**

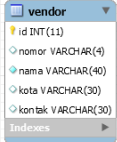
- Problem: menampilkan data pelanggan yang lahir di Jakarta



- Result: 2 records

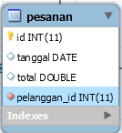


### **SQL SELECT : Update, Delete**

- Problem: mengubah data kota dari table vendor nomor V003 menjadi   
 kota Purwerejo



- Result: 1 record ter-update

- Problem: menghapus data pesanan untuk pelanggan\_id =14



- Result: 4 records ter-hapus

### **SQL SELECT : Order By**

- Problem: menampilkan seluruh data diurutkan berdasarkan nama secara alfabet



- Catatan : Default order alfabet dari A – Z, DESC dari Z – A

- Result: 14 records



- Problem: menampilkan seluruh data pelanggan yang lahir di kota Jakarta, Bogor, Kediri dan mengurutkan berdasarkan nama pelanggan



- Result: 5 records



### **Klausa DISTINCT**

- Keyword DISTINCT digunakan untuk menampilkan informasi data suatu kolom yang bernilai berbeda

- data yang sama akan dimunculkan satu kali saja

Tampilkan kota-kota tempat kelahiran

- SELECT DISTINCT(tmp\_lahir) as kota\_kelahiran FROM pelanggan

### **SQL SELECT : DISTINCT**

- Problem: Tampilkan data kota tempat lahir pelanggan diurutkan berdasarkan kota kelahiran



- Results: 10 records

### **Operator Perbandingan**

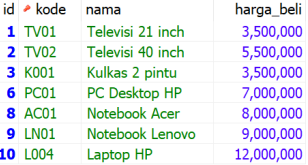
Operator perbandingan

* = sama dengan
* < lebih kecil dari
* <= lebih kecil dari atau sama dengan
* > lebih besar dari
* >= lebih besar dari atau sama dengan
* < > , != tidak sama dengan

#### **SQL SELECT : AND , OR , NOT**

- Problem: Tampilkan data kode,nama dan harga\_beli dari table produk yang harganya diatas 3juta

- Results: 7 records



- Problem: Tampilkan data produk yang ID nya 3 , 5 dan 7



- Results: 3 record



- Problem: Tampilkan data nama,jk,tmp\_lahir,tgl\_lahir dari pelanggan yang tmp\_lahirnya bukan di Jakarta dan kendiri



- Result: 10 record



### **SQL SELECT : Between**

- Problem: Tampilkan data produk yang harganya antara 5juta sampai 10 juta



- Result: 4 records



### **Keyword LIKE dan NOT LIKE**

Digunakan untuk pencarian data string

- Tampilkan data produk yang nama produk berawal ‘ k ‘

* SELECT \* FROM produk WHERE nama LIKE ‘k%’;

- Tampilkan data produk yang nama produk memiliki akhiran ‘ an ‘

* SELECT \* FROM produk WHERE nama LIKE ‘%an’ ;

- Tampilkan data produk yang nama produk memiliki suku kata ‘ gu ‘

* SELECT \* FROM produk WHERE nama LIKE ‘%gu%’ ;

**Like**

- Problem: Tampilkan data kode,nama dan harga\_beli dari table produk yang nama awalnya ‘Te’



- Result: 4 records



### **Klausa LIMIT dan OFFSET**

Klausa LIMIT digunakan untuk menampilkan sejumlah baris data dari table, sedangkan OFFSET untuk menentukan baris awal data yang akan ditampilkan

- Tampilkan 3 data pertama dari table produk

* SELECT \* FROM produk LIMIT 3

- Tampilkan 3 data produk dimulai dari data ke 3

* SELECT \* FROM produk LIMIT 3 OFFSET 2

Contoh:

* Problem: tampilkan 3 data pelanggan mulai dari baris ke 5



* Result: 2 records



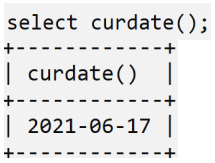
# **PERTEMUAN 07 Perintah SQL Select 2 Built-in Function**

## **SQL Data Type : Date & Time**

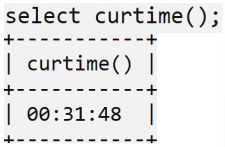
* DATE – format: YYYY-MM-DD
* DATETIME – format: YYYY-MM-DD HH:MI:SS
* TIMESTAMP – format: YYYY-MM-DD HH:MI:SS
* YEAR – format: YYYY or YY

### **Fungsi Waktu**

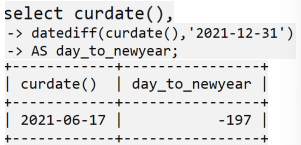
* **Curdate()** untuk menampilkan tahun, bulan dan tanggal



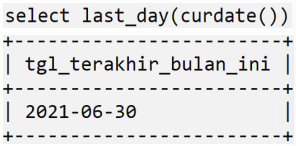
* **Curtime()** untuk menampilkan waktu



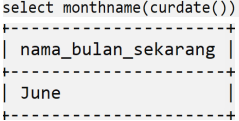
* **Datediff (curdate(), ‘2022-12-31’)** menampilkan berapa hari sebelum tahun baru



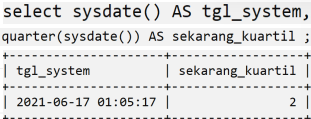
* **Last\_day(curdate())** menampilkan tanggal terakhir di bulan ini



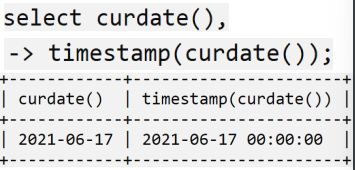
* **Monthname(curdate())** menampilkan nama bulan sekarang



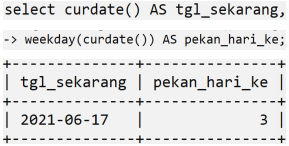
* **Sysdate()** menampilkan tanggal dan jam, **quarter(sysdate())** menampilkan kuartil



* **Timestamp(curdate))** digunakan untuk dapat mengembalikan nilai datetime berdasarkan tanggal atau nilai datetime



* **Weekday(curdate())** menampilkan pekan ke berapa

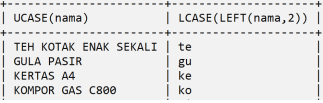


### **Fungsi String**

* **Concat** untuk menampilkan string / text

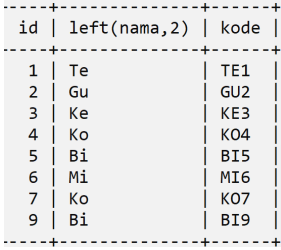


* **Ucase/lcase** untuk menampilkan string



* RPAD / Right-Pads





Fungsi Control Flow

