ERD + Functional Dependency

Modul Praktikum Sistem Manajemen Basis Data 2025

Tim Asisten
Sistem Manajemen
Basis Data 2025

Rollback teori di kelas

Karena teman-teman nggak ada pertemuan SMBD Teori, maka dari itu Praktikum difokuskan untuk Refresh Frame Rate otak, dengan Materi-materi SMBD yang sama, namun dengan pembahasan yang lebih advance. Iya ini yapping.

Di Kesempatan ini, bakalan ada pengenalan materi yang akan dijelaskan pada pertemuan selanjutnya.

Sistem Berkas dan Permasalahannya

Ada 7 masalah pokok dalam Sistem Berkas Konvensional:

- **Data Redundancy and Inconsistency**
  c/ Sistem manual, informasi disimpan dalam **beberapa file** tersimpan. Terjadi Update di 1 file, namun tidak dilakukan di file lain.

- **Difficulty in Accessing Data**
  c/ Saat mengakses data transaksi dalam rentang 1 tahun, namun informasi tersimpan dalam beberapa file terpisah tanpa penanda yang jelas.

- **Data Isolation**
  c/ Data stok barang disimpan dalam 1 file, Data Transaksi disimpan dalam file lain tanpa ada keterhubungan. Jika ingin mengecheck barang harus dicocokkan secara manual.

- **Integrity Problems**
  c/ Ada Aturan Bahwa Mahasiswa harus membayar UKT untuk bisa mengontrak IRS, tetapi SIAK tidak memiliki mekanisme untuk memastikan UKT sudah dibayar atau belum. Bisa jadi Mahasiswa mengontrak tanpa bayar UKT.

- **Atomicity Problems**
  c/ Seorang Mahasiswa mengirim uang ke orang tuanya melalui transfer bank, Di Sistem muncul notifikasi saldo sudah dipotong, namun bukti transaksi tidak pernah muncul.

- **Concurrent Access-Anomalies**
  c/ Dua pelanggan mengedit data di google spreadsheet secara bersamaan, dan sistem tidak memiliki mekanisme penguncian file, maka salah satu perubahan bisa tertimpa dengan perubahan lainnya tanpa disadari.

- **Security Problems**
  c/ Jika semua karyawan memiliki akses langsung ke file keuangan perusahaan tanpa adanya sistem granted access, maka ada risiko bahwa data keuangan bisa dirubah oleh orang yang tidak berwenang.

Perbedaan Data & Informasi

- **Data** : Fakta Dunia nyata yang dapat didefinisikan berupa nilai numerik, alfanumerik, gambar, suara, etc.
- **Informasi** : Hasil pengelolaan dari data yang memiliki makna bagi penggunanya.

Tujuan Penggunaan Basis Data

Mempermudah dan Mempercepat Proses pengolahan Data

- Mengontrol **Redundansi Data**
- Menjaga **Konsistensi Data**
- Pemakaian Data Dapat **Dilakukan Secara Bersamaan**
- **Meningkatnya Integritas Data**
- Meningkatnya **Keamanan Database**
- Mengadopsi Sistem **User Oriented**

Data Definition Language (DDL)

Data Definition Language adalah Command yang digunakan untuk **membangun** skema Basis Data. Contohnya:

- CREATE TABLE Mahasiswa..
- ALTER TABLE Mahasiswa..
- DROP TABLE Mahasiswa..
- TRUNCATE TABLE Mahasiswa..

Data Manipulation Language (DML)

Data Manipulation Language berisi Command yang digunakan untuk **memanipulasi** skema Basis Data. Contohnya:

- INSERT INTO Dosen..
- SELECT * FROM Dosen..
- UPDATE Dosen..
- DELETE FROM Dosen..

Bahasa Model Relational

Query Language adalah istilah lain dalam DML, digunakan untuk melakukan Pencarian, Pemilihan, sampai Penayangan Data.

Contoh Command SQL :

SELECT * FROM Dosen Terbaik WHERE Rating_Dosen == A;

Terjemahan dalam **Kalkulus Relasional**

$$\{ x \mid x \in \text{DOSEN\_TERBAIK} \wedge x\,[\text{Rating\_Dosen}] = \text{'A'} \}$$

Ket:

- x adalah himpunan tuple dari tabel **DOSEN_TERBAIK**.
- Kita hanya memilih tupel yang memiliki **Rating_Dosen = 'A'**.

Terjemahan dalam **Aljabar Relasional**

$$\sigma_{\text{Rating\_Dosen} = \text{'A'}}(\text{DOSEN\_TERBAIK})$$

Ket:

- **σ (seleksi)** digunakan untuk memilih baris dengan kondisi **Rating_Dosen = 'A'**.
- Operasi ini dilakukan pada relasi **DOSEN_TERBAIK**.

Kenapa Harus bikin ERD di SMBD?

Di Sistem Manajemen Basis Data, kita dituntut untuk **belajar kreatif dalam mendesain** keterhubungan antara satu data, dengan yang lainnya.

Jika kita membuat tabel-tabel, menyambungkan database tanpa membuat rancangan diagram terlebih dahulu, **Bagaimana cara kita mengembangkan database, bila kita tidak tahu gambaran besar relasi database itu sendiri.**

Entitas dan Atribut

Entitas : Sesuatu yang nyata dan bisa dikenali dalam dunia nyata (database)
c/ **Superhero**, Mobil, Siswa di sekolah

Atribut : Hal-Hal yang menjelaskan entitas.
c/ kekuatan, warna Kostum, asal kota

Contoh Penerapannya:

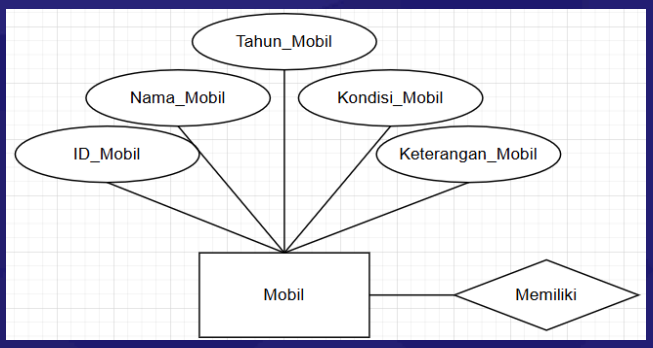
Entitas Superhero

Nama Superhero	Kekuatan	Warna Kostum	Asal Kota
Spider_Man	Memanjat Dinding	Merah_Biru	New_York
Batman	Kaya	Hitam	Gotham_City
Gundala	Secepat Kilat	Merah_Maroon	Jakarta

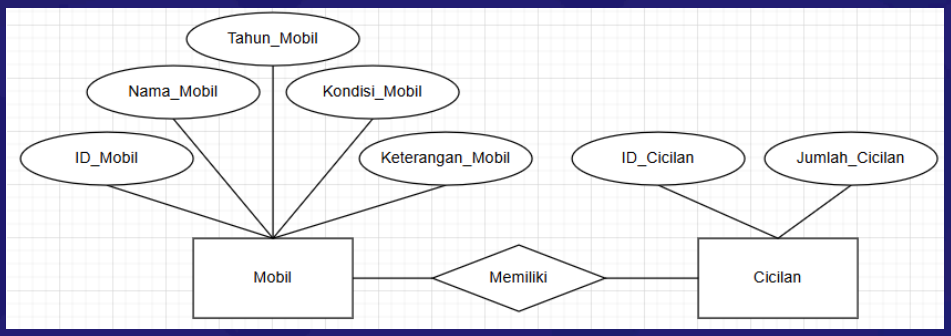
Relasi

Relasi Fungsinya untuk menggambarkan hubungan antar entitas agar sesuai dengan proses bisnisnya. Biasanya digambarkan dalam notasi Belah Ketupat.

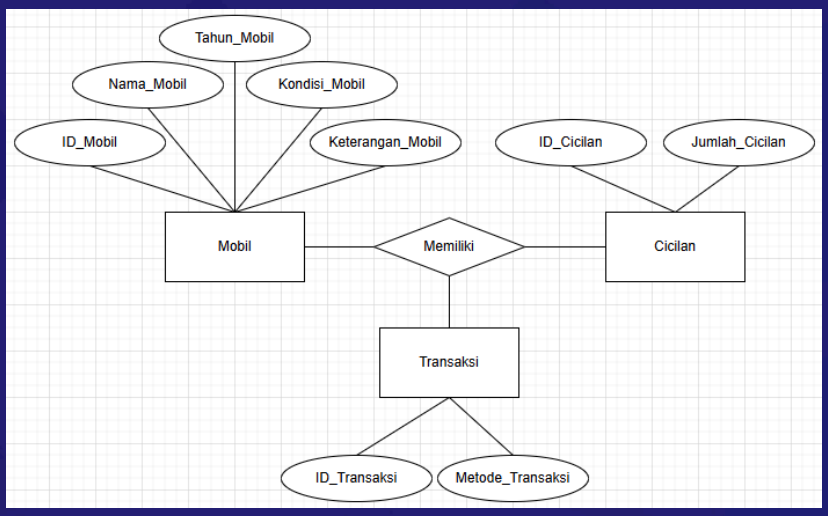
Unary (Satu Bagian)



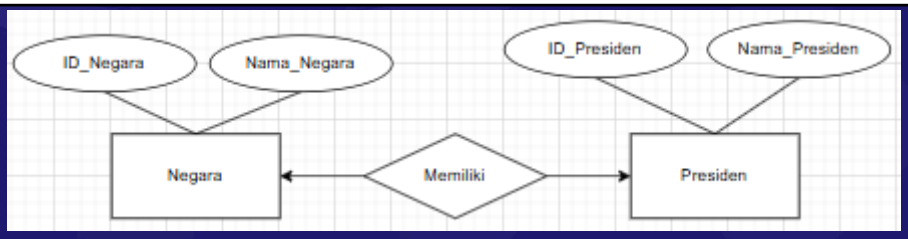
Binary (Dua Bagian)



Ternary (Lebih dari Dua Bagian)

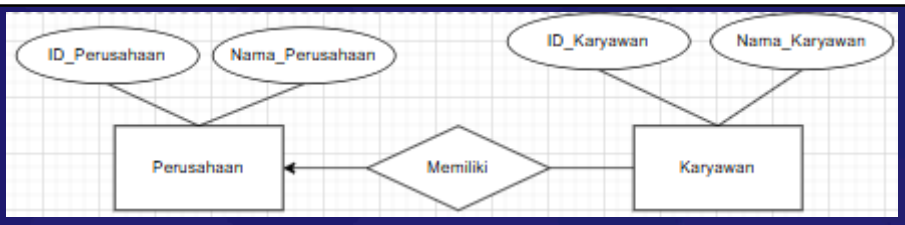


Kardinalitas One to One



a. Setiap Negara hanya memiliki satu presiden dalam satu waktu
b. Setiap Presiden hanya menjabat di satu negara

Kardinalitas One to Many

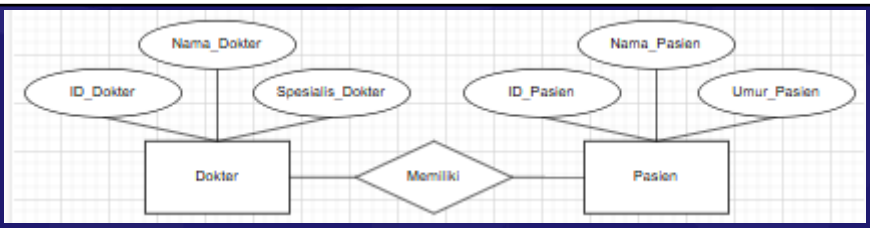


a. Satu **Perusahaan** memiliki banyak **Karyawan**
b. Tapi setiap **Karyawan** hanya bekerja di satu **Perusahaan**

Kardinalitas Many to One

a. Banyak **Buku** bisa diterbitkan oleh satu **Penerbit**
b. Tapi setiap **Buku** hanya memiliki satu **Penerbit**

Kardinalitas Many to Many



a. Satu **Dokter** bisa menangani Banyak **Pasien**
b. Satu **Pasien** bisa ditangani oleh banyak **Dokter** (misalnya, Pasien yang dirawat oleh dokter spesialis dan dokter umum

Functional Dependency

Cara paling sederhana untuk menelaah Functional Dependency adalah dengan menerapkan studi kasus dalam kejadian nyata:

ID_Pegawai	Nama	Jabatan	Departemen	Gaji
601	Neuvillette	Gubernur	Fontaine	10 Juta
602	Furina	Gubernur	Fontaine	10 Juta
603	Miko	Pendeta	Inazuma	9 Juta
604	Chasca	Bodyguard	Natlan	7 Juta
605	Xilonen	Pengrajin	Natlan	9 Juta

a. Functional Dependency

FD berarti nilai suatu atribut bisa menentukan nilai atribut lain.
Contoh dari Functional Dependency:

- **ID_Pegawai -> Nama, Jabatan, Departemen, ID_Departemen, Gaji** (Karena setiap pegawai memiliki ID unik, maka ID_Pegawai dapat menentukan atribut lainnya)
- **ID_Departemen → Departemen** (Karena ID Departemen bersifat unik untuk setiap departemen).

b. Super Key

Super Key adalah satu atau lebih atribut yang dapat mengidentifikasi satu baris dalam tabel secara unik.

Contoh dari Super Key

- {**ID_Pegawai**} -> Karena ID Pegawai Unik
- {**ID_Pegawai, Nama**} -> Bisa juga, tetapi sudah berlebihan karena ID_Pegawai saja sudah cukup
- {**ID_Pegawai, Jabatan, Departemen**} -> juga bisa, tetapi tetap tidak efisien

c. Candidate Key

Candidate Key adalah Super Key yang minimal (tidak ada atribut tambahan yang berlebihan)

Contoh Candidate Key:

- {**ID_Pegawai**} -> Karena bisa mengidentifikasi setiap pegawai tanpa atribut tambahan

d. Primary Key

Primary Key adalah Candidate Key yang dipilih untuk menjadi pengenal utama dalam tabel

Contoh Primary Key

- **ID_Pegawai** (karena untuk untuk setiap pegawai)

e. The Keys

Jenis Key	Contoh
Super Key	{ID_Pegawai}, {ID_Pegawai, Nama}
Candidate Key	{ID_Pegawai}
Primary Key	ID_Pegawai

f. Partial Dependency

Misalnya kita anggap (**ID_Pegawai, Jabatan**) sebagai kunci utama

a. **Nama** hanya bergantung pada **ID_Pegawai**, bukan pada kombinasi (**ID_Pegawai, jabatan**).
b. Artinya, jika kita tahu **ID_Pegawai**, kita bisa langsung tahu **Nama** pegawai tersebut, tanpa perlu melihat jabatan.

c. Ini disebut dengan **Partial Dependency**, karena **Nama** hanya bergantung pada sebagian kunci utama (**ID_Pegawai**) saja.

Tips : Seperti kalau kamu tahu nomor pegawai seseorang, kamu langsung tahu namanya. Kamu nggak perlu tahu jabatan mereka untuk mengetahuinya.

g. Transitive Dependency

Misalnya kita tahu kalau:

a. Jabatan menentukan Departemen (Misalnya, kalau Gubernur pasti ada di Fontaine)

b. Departemen selalu menentukan besar gaji (Misalnya, kalau di Fontaine pasti Gajinya 10 Juta)

Jadi, **Jabatan -> Departemen -> Lokasi Kantor** adalah **Transitive Dependency.**

Tips:
Kalau kamu tahu seseorang adalah "Gubernur", otomatis dia ada di Fontaine, dan kalau dia di Fontaine, berarti dia gajinya 10 Juta.

Pro Tips untuk Functional Dependency:

- Untuk menghindari Partial Dependency, kita bisa buat tabel PEGAWAI (ID_Pegawai, Nama, Jabatan) dan tabel JABATAN (Jabatan, Departemen, Lokasi_Kantor).
- Untuk menghindari Transitive Dependency, kita bisa pisahkan Departemen dan Lokasi Kantor ke tabel lain.

ERD

Langsung Praktek aja :0

Tugas

To Be Announced

Penutup

Kami mengucapkan terima kasih kepada Tuhan, Bangsa, dan Almamater. Yang telah berkontribusi dalam pembuatan modul ini. Semoga pengetahuan dan pengalaman yang diperoleh dapat memberikan manfaat yang signifikan bagi masa depan. Aamiin.