# Model Relasional: Fondasi Penyimpanan Data Modern dan Transformasi ERD ke Tabel

**Muhammad Riduwan Khafidi**
Program Studi Teknik Rekayasa Perangkat Lunak
Politeknik Negeri Batam
4342611034@students.polibatam.ac.id

November 2, 2026

**Abstract**

Model relasional yang diperkenalkan oleh Edgar F. Codd pada tahun 1969 merupakan tonggak sejarah dalam pengelolaan basis data. Model ini merevolusi cara data disimpan dan diakses dengan memetakan seluruh informasi ke dalam bentuk tabel dua dimensi yang sederhana, namun memiliki landasan matematis yang kuat. Makalah ini membedah secara komprehensif konsep dasar model relasional, mulai dari sejarah, definisi, terminologi formal (seperti relasi, tuple, dan atribut), karakteristik tabel yang valid, hingga hirarki kunci (keys) yang menjamin integritas data. Bagian utama makalah ini berfokus pada panduan sistematis untuk menerjemahkan skema konseptual (Entity Relationship Diagram) menjadi skema fisik (Relational Schema). Pembahasan mencakup pemetaan entitas, relasi (one-to-one, one-to-many, many-to-many), atribut multivalued, agregasi, hingga strategi penanganan spesialisasi (inheritance).

## 1 Pendahuluan: Revolusi Edgar F. Codd

Dalam alur perancangan basis data, setelah kita berhasil memetakan kebutuhan pengguna ke dalam model konseptual (seperti ERD atau EERD), tahap selanjutnya adalah menerjemahkannya ke dalam model fisik. Di sinilah Model Relasional berperan. Dirumuskan pertama kali oleh Edgar F. Codd (seorang ilmuwan komputer di IBM) pada tahun 1969, model ini mengubah cara pandang dunia terhadap basis data.

Sebelum adanya model relasional, basis data dikelola menggunakan model hierarkis atau jaringan yang sangat rumit. Codd mengajukan pendekatan yang jauh lebih elegan: seluruh data diwakili dalam bentuk tabel dua dimensi (yang secara matematis disebut sebagai *relations*). Pendekatan ini tidak hanya memudahkan para programmer dalam melakukan operasi manipulasi data (seperti *query*, *update*, dan *delete*), tetapi juga sangat mudah dipahami bahkan oleh kalangan non-programmer karena bentuknya yang intuitif.

## 2 Terminologi Dasar Model Relasional

Meskipun dalam percakapan sehari-hari kita terbiasa menggunakan istilah "tabel", "baris", dan "kolom", model relasional memiliki terminologi formal yang wajib dipahami. Gambar 1 mengilustrasikan padanan antara istilah formal dengan istilah alternatif yang umum digunakan.

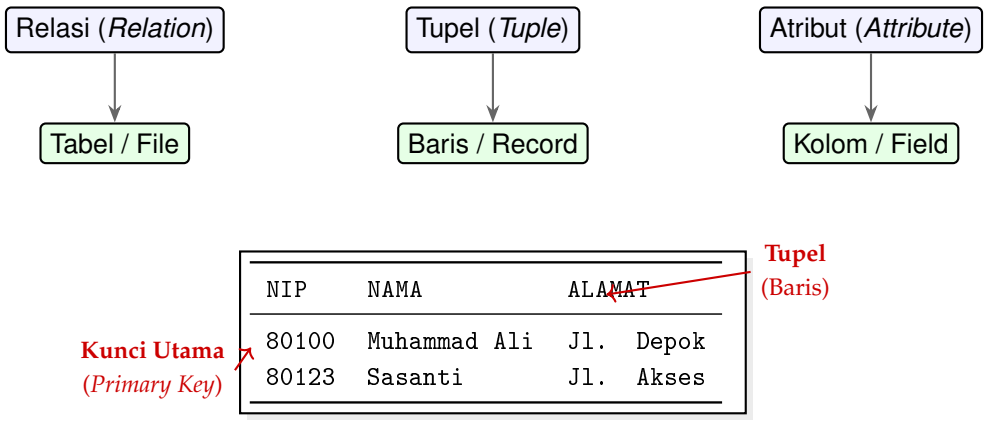



**Figure 1.** Terminologi Formal Model Relasional dan Contoh Penerapannya.

Secara garis besar, sebuah baris data disebut sebagai **Tupel**. Kumpulan dari nilai-nilai yang mungkin untuk sebuah atribut (misalnya kode divisi: Div01, Div02) disebut **Domain**. Jumlah kolom dalam sebuah tabel disebut **Degree** (Derajat), sedangkan jumlah baris dalam tabel disebut **Cardinality** (Kardinalitas).

## 3 Karakteristik dan Aturan Tabel yang Valid

Tidak semua tabel bisa disebut sebagai relasi yang valid dalam model ini. Terdapat aturan ketat yang harus dipenuhi. Pertama, setiap sel dalam tabel harus bersifat **atomik**, yang berarti nilainya tidak boleh dipecah lagi menjadi beberapa bagian. Sebagai contoh, kolom "Alamat" tidak boleh berisi "Jl. Ahmad Yani No. 5, Batam", karena itu terdiri dari beberapa nilai (jalan, nomor, kota). Jika ingin atomik, harus dipisah menjadi kolom Jalan, Nomor, dan Kota.

Kedua, tidak boleh ada dua tupel (baris) yang identik. Ketiga, setiap atribut (kolom) harus memiliki nama yang unik. Keempat, urutan baris maupun kolom tidak memiliki arti khusus (tidak harus tersusun secara khusus).

## 4 Hirarki Relational Key

Untuk memastikan tidak ada data yang duplikat, kita membutuhkan Kunci (*Key*). Dalam model relasional, terdapat hirarki kunci yang penting untuk dipahami:

Sebuah **Super Key** adalah satu atau kumpulan atribut yang secara unik dapat mengidentifikasi sebuah tupel. Dari super key ini, kita memilih yang paling minimal (tanpa atribut yang mubazir) yang disebut **Candidate Key**. Dari beberapa kandidat, kita memilih satu yang paling

utama sebagai **Primary Key** (Kunci Utama). Sisa kandidat yang tidak terpilih disebut sebagai **Alternate Key**. Terakhir, sebuah atribut yang menjadi Primary Key di tabel lain, namun hanya menjadi atribut biasa di tabel ini, disebut sebagai **Foreign Key** (Kunci Tamu).

## 5 Transformasi ER-Diagram ke Skema Relasional

Ini adalah inti dari praktikum pertemuan ini. Bagaimana kita mengubah gambar konseptual (ERD) menjadi tabel fisik? Berikut adalah aturan baku yang harus diikuti.

### 5.1 Entitas Kuat dan Entitas Lemah

Setiap entitas kuat (misalnya "Pegawai") langsung diterjemahkan menjadi sebuah tabel, dengan atribut-atributnya menjadi kolom, dan Primary Key-nya menjadi kunci utama tabel tersebut.

Adapun untuk entitas lemah (misalnya "Pembayaran" yang bergantung pada "Pinjaman"), kita membuat satu tabel tersendiri. Primary Key untuk tabel entitas lemah adalah gabungan dari Primary Key entitas kuat ditambah dengan *partial key* (discriminator) miliknya sendiri. Primary Key entitas kuat ini sekaligus berfungsi sebagai Foreign Key.

### 5.2 Relasi One-to-Many (1:N) dan One-to-One (1:1)

Aturan paling efisien untuk relasi One-to-Many adalah dengan menambahkan Primary Key dari sisi "One" sebagai Foreign Key ke tabel di sisi "Many".

Sebagai ilustrasi, perhatikan Gambar 2. Terdapat relasi "Manager" antara entitas "Project" dan "Employee", di mana satu Project hanya punya satu Manager, tetapi satu Employee bisa mengelola banyak Project. Maka, kita menambahkan atribut ID Manager (Foreign Key) ke dalam tabel Project.




**Figure 2.** Transformasi Relasi One-to-Many menjadi Foreign Key.

**Project** (P-number, P-name, Due-Date, *Manager*)
**Employee** (SSN, E-name, Office)

Jika relasi tersebut memiliki atribut (misalnya tanggal mulai menjadi manager), maka atribut tersebut ikut dipindahkan ke tabel di sisi "One" bersama dengan Foreign Key-nya. Adapun untuk relasi One-to-One, aturannya mirip, kita bebas menaruh Foreign Key di salah satu tabel (biasanya di tabel yang paling sering diakses).

### 5.3 Relasi Many-to-Many (M:N)

Relasi Many-to-Many tidak bisa diselesaikan hanya dengan menambah Foreign Key di salah satu tabel. Kita **wajib membuat tabel baru** (sering disebut tabel asosiasi/junction). Tabel baru ini setidaknya memiliki dua kolom yang merupakan Foreign Key dari kedua entitas yang dihubungkan, dan kedua kolom tersebut dijadikan satu kesatuan sebagai Primary Key komposit.

Sebagai contoh, pada relasi "Assignment" antara Project dan Employee, kita buat tabel baru bernama Assignment.




**Figure 3.** Pembuatan Tabel Baru untuk Relasi Many-to-Many.

**Project** (P-number, P-name, Due-Date)
**Employee** (SSN, E-name, Office)
**Assignment** (P-number, SSN)

### 5.4 Atribut Multivalued dan Komposit

Atribut yang memiliki lebih dari satu nilai (seperti "Nomor Telepon" pada Pegawai) tidak diizinkan dalam model relasional (melanggar aturan atomik). Solusinya, kita harus membuat tabel terpisah untuk menampungnya. Tabel baru tersebut berisi Primary Key entitas asal ditambah dengan atribut multivalued itu sendiri.

Adapun atribut komposit (seperti "Alamat" yang terdiri dari Jalan dan Kota) cukup dipecah (di-*flatten*) menjadi kolom-kolom terpisah di dalam tabel yang sama.

### 5.5 Agregasi dan Spesialisasi (EERD)

Pada topik EERD sebelumnya, kita mempelajari Spesialisasi (Pewarisan). Terdapat tiga strategi untuk mengubahnya menjadi tabel:

Pertama, pendekatan **ER-Style**, di mana kita membuat tabel terpisah untuk *superclass* dan setiap *subclass*. Tabel *subclass* hanya berisi Primary Key dari *superclass* ditambah atribut uniknya.

Kedua, pendekatan **OO-Style** (Object-Oriented), di mana setiap *subclass* dibuatkan tabel tersendiri yang memuat *semua* atribut (baik dari *superclass* maupun atribut uniknya sendiri). Tabel *superclass* mungkin tidak diperlukan jika sering diakses dari *subclass*.

Ketiga, pendekatan **NULL Value**, di mana semua entitas digabung ke dalam satu tabel besar (Single Table) yang berisi semua atribut dari seluruh *subclass*. Atribut yang tidak dimiliki oleh *subclass* tertentu akan dibiarkan bernilai *NULL* (kosong).

Terakhir, untuk relasi Agregasi (seperti Customer meminjam Loan, lalu membayar Payment), transformasinya sama seperti Many-to-Many, yaitu dibuatkan tabel baru yang memuat Foreign Key dari entitas-entitas yang terlibat dalam agregat tersebut.

## 6 Kesimpulan

Model Relasional adalah sebuah mahakarya yang menyederhanakan kompleksitas data menjadi bentuk tabel yang terstruktur. Melalui aturan formalisasi yang ketat (seperti domain, derajat, kardinalitas, dan kunci), integritas data dapat terjaga dengan sempurna. Kemampuan seorang perancang basis data untuk menerjemahkan ER-Diagram (konsep) menjadi Skema Relasional (fisik) secara akurat merupakan keterampilan esensial. Pemetaan yang tepat akan memastikan bahwa sistem basis data yang dibangun nantinya tidak hanya bebas dari redundansi, tetapi juga memiliki performa yang optimal saat dieksekusi melalui *query* SQL.

## Referensi

## References

[1] Lubis, A. I. (2026). *Relational Model* [Slide Presentasi Mata Kuliah Pengantar Basis Data]. Politeknik Negeri Batam.

[2] Silberschatz, A., Korth, H. F., & Sudarshan, S. (2019). *Database System Concepts* (7th ed.). McGraw-Hill Education.

[3] Santiputri, M. (2012). *Diktat Pengantar Basis Data*. Politeknik Negeri Batam.