NORMALISASI, KUNCI RELASIONAL DAN TIPE DATA

A. Normalisasi

Normalisasi adalah proses mengorganisasi struktur tabel dalam basis data agar data menjadi lebih efisien, tidak *redundan* (berulang), dan mudah dikelola.

Tujuan utamanya adalah untuk:

- Menghilangkan pengulangan data (*redundansi*);
- Meningkatkan konsistensi data (integritas data);
- Memudahkan pemeliharaan basis data.

Ciri Proses Normalisasi:

- Memecah tabel besar menjadi beberapa tabel yang saling berhubungan.
- Mengidentifikasi dan menggunakan primary key serta foreign key.
- Mengelompokkan data ke dalam tabel sesuai entitasnya (contoh: siswa, mapel, nilai, dll).

Studi Kasus: Data Nilai Siswa

Perhatikan data nilai (belum normal berikut)

NIS	Nama Siswa	Kelas	Mapel 1	Nilai 1	Mapel 2	Nilai 2
101	Ana	X 1	Matematika	85	Fisika	80
102	Budi	X 1	Matematika	90	Fisika	88

Sepintas tabel data nilai siswa tampak wajar, tapi sesungguhnya tabel data nilai siswa tersebut memiliki sejumlah masalah. Data awal yang masih belum normal seperti ini dalam basis data disebut tabel tidak normal / *Unnormalized Form* **(UNF)**. UNF adalah bentuk awal dari struktur tabel dalam basis data sebelum dilakukan proses normalisasi. Data dalam bentuk UNF biasanya masih belum terorganisir dengan baik, dan mengandung *data ganda, data bertingkat, atau data berulang dalam satu kolom atau baris*.

Ciri-ciri UNF

- Satu kolom bisa berisi lebih dari satu nilai (*multivalue*).
- Data masih *redundan* atau *duplikatif*
- Tidak ada struktur yang jelas antara entitas dan atribut.
- Belum ditentukan *primary key*.

Masalah yang terdapat pada tabel Data Nilai diatas adalah:

- Satu baris menyimpan dua nilai mapel → duplikasi struktur
- Tidak bisa menyimpan lebih dari dua mapel tanpa menambah kolom baru
- Redundansi data mapel

Proses Normalisasi

1. **1NF (*First Normal Form*):**

   Hilangkan pengelompokan data dalam satu kolom. Setiap nilai harus atomik.

NIS	Nama Siswa	Kelas	Mapel	Nilai
101	Ana	X 1	Matematika	85
101	Ana	X 1	Fisika	80
102	Budi	X 1	Matematika	90
102	Budi	X 1	Fisika	88

2. **2NF (*Second Normal Form*):**

   Hilangkan redundansi data yang bergantung pada bagian dari kunci utama (*primary key*) dan pisahkan ke dalam beberapa tabel.

   a) **Tabel: Siswa**

NIS	Nama Siswa	Kelas
101	Ana	X 1
102	Budi	X 1

   b) **Tabel: Mapel**

Kode_Mapel	Nama Mapel
M01	Matematika
M02	Fisika

   c) **Tabel: Nilai**

NIS	Kode_Mapel	Nilai
101	M01	85
101	M02	80
102	M01	90
102	M02	88

3. **3NF (***Third Normal Form***):**

   Pastikan tidak ada ketergantungan transitif, yaitu kolom non-kunci bergantung pada kolom non-kunci lainnya.

   Dalam studi kasus ini, sudah memenuhi 3NF, karena:

   - Semua data non-kunci bergantung langsung pada kunci utama (*primary key*) masing-masing tabel

B. Kunci Relasional (*Primary Key dan Foreign Key*)

Dalam proses normalisasi, kita memecah tabel besar menjadi beberapa tabel yang lebih kecil. Agar data tetap bisa terhubung, kita perlu *primary key* dan *foreign key*.

Primary Key (Kunci Utama)

Primary Key adalah kunci utama dalam sebuah tabel basis data yang digunakan untuk mengidentifikasi setiap baris data secara unik. Kolom yang dijadikan primary key tidak boleh kosong (NULL) dan tidak boleh memiliki nilai yang sama antar baris.

Ciri-ciri Primary Key:

- Nilainya unik untuk setiap baris
- Wajib diisi (tidak boleh NULL)
- Umumnya digunakan untuk menghubungkan antar tabel melalui foreign key

Contoh:

- NIS di tabel siswa
- kode_mapel di tabel mapel

Foreign Key (Kunci Tamu)

Kunci Tamu *(Foreign Key)* adalah kolom dalam sebuah tabel yang digunakan untuk menghubungkan tabel tersebut dengan tabel lain melalui kolom yang menjadi *Primary Key* di tabel tujuan. Dengan kata lain, *foreign key* adalah cara untuk menjaga hubungan antar tabel dalam basis data relasional.

Ciri-ciri Foreign Key:

- Mengacu (mereferensikan) ke primary key di tabel lain.
- Bisa memiliki nilai berulang, tergantung relasinya.
- Bisa berisi NULL (jika relasinya tidak wajib).
- Digunakan untuk menjaga integritas data antar tabel.

Contoh:

- NIS dan kode_mapel di tabel nilai adalah foreign key yang mengarah ke tabel siswa dan mapel.

C. Tipe Data

Secara umum, database relasional memiliki Tipe Data Dasar untuk perancangan tabel yang meliputi:

Tipe Data	Kegunaan
INT	Angka bulat, misalnya NIS, kode mapel
VARCHAR(n)	Teks pendek, maksimal n karakter (misalnya 100)
YEAR	Tahun (misal untuk tahun masuk)
DATE	Format tanggal lengkap (YYYY-MM-DD)
FLOAT/DECIMAL	Angka desimal, misalnya nilai dengan koma

Secara spesifik tipe data pada database relasional MySQL/MariaDB dapat dijabarkan sebagai berikut:

1. Tipe Data Numerik

Digunakan untuk menyimpan nilai angka, terbagi menjadi:

Tipe Data	Ukuran/Format	Range Nilai	Contoh Penggunaan
TINYINT	1 byte	-128 s/d 127 (signed) / 0-255 (unsigned)	Status aktif (0/1)
INT	4 byte	-2^{31} s/d $2^{31}-1$ (signed)	ID user, jumlah stok
BIGINT	8 byte	-2^{63} s/d $2^{63}-1$ (signed)	Nomor transaksi besar
FLOAT	4 byte (presisi tunggal)	±1.18E-38 s/d ±3.40E+38	Nilai ilmiah (e.g., suhu)
DECIMAL(p,s)	Variabel (p=total digit, s=desimal)	Presisi tetap	Harga produk (e.g., DECIMAL(10,2))

Catatan:

- UNSIGNED untuk nilai non-negatif (contoh: INT UNSIGNED).
- ZEROFILL menambahkan leading zero (e.g., INT(5) ZEROFILL → 0042).

2. Tipe Data String (Teks)

Untuk menyimpan data karakter:

Tipe Data	Maksimal Ukuran	Karakteristik	Contoh Penggunaan
CHAR(n)	255 karakter	Fixed-length, lebih cepat	Kode negara (e.g., 'ID', 'US')
VARCHAR(n)	65,535 karakter	Variable-length, hemat storage	Nama user, alamat email
TEXT	65,535 karakter	Untuk teks panjang	Deskripsi produk
ENUM	65,535 nilai unik	Hanya pilihan tertentu	Jenis kelamin ('Male', 'Female')

Perbedaan CHAR vs VARCHAR:

- CHAR(10) selalu pakai 10 byte (diisi spasi jika kurang).

- VARCHAR(10) hanya pakai 1 byte + panjang teks (e.g., 'ABC' = 4 byte).

---

3. Tipe Data Tanggal & Waktu

Menyimpan informasi temporal:

Tipe Data	Format	Range Nilai	Contoh Penggunaan
DATE	YYYY-MM-DD	1000-01-01 s/d 9999-12-31	Tanggal lahir
DATETIME	YYYY-MM-DD HH:MM:SS	1000-01-01 00:00:00 s/d 9999-12-31 23:59:59	Log aktivitas
TIMESTAMP	YYYY-MM-DD HH:MM:SS	1970-01-01 00:00:01 UTC s/d 2038-01-19 03:14:07 UTC	Auto-update (e.g., ON UPDATE CURRENT_TIMESTAMP)

Perbedaan DATETIME vs TIMESTAMP:

- TIMESTAMP konversi ke zona waktu server, DATETIME tidak.
- TIMESTAMP butuh 4 byte, DATETIME 8 byte.

4. Tipe Data Binary

Untuk data biner (e.g., file/gambar):

Tipe Data	Maksimal Ukuran	Kegunaan
BLOB	65,535 byte (TINYBLOB) s/d 4GB (LONGBLOB)	Gambar, PDF
BINARY(n)	n byte (maks 255)	Data biner fixed-length

5. Tipe Data Spasial (GIS)

Khusus untuk data geografis:

- **GEOMETRY**: Menyimpan semua tipe geometri.
- **POINT**: Koordinat (latitude, longitude).

Implementasi Tipe Data pada Struktur Tabel Hasil Normalisasi

Tabel siswa

Nama Kolom	Tipe Data	Keterangan
nis	INT	PRIMARY KEY
nama	VARCHAR(100)	
kelas	VARCHAR(10)	

Tabel mapel

Nama Kolom	Tipe Data	Keterangan
kode_mapel	VARCHAR(10)	PRIMARY KEY
nama_mapel	VARCHAR(100)	

Tabel nilai

Nama Kolom	Tipe Data	Keterangan
nis	INT	FOREIGN KEY → siswa.nis
kode_mapel	VARCHAR(10)	FOREIGN KEY → mapel.kode_mapel
nilai	INT	

Kesimpulan

- Normalisasi membantu membuat struktur database **terorganisir**.
- Primary key dan foreign key membuat **relasi antar tabel** dapat diakses dengan efisien.
- Pemilihan **tipe data yang tepat** membantu menyimpan data dengan akurat dan hemat ruang/memori.