

PERANCANGAN BASIS DATA (PBD)

3 SKS | Semester 5 | S1 Sistem Informasi | UNIKOM | 2014

Nizar Rabbi Radliya | nizar.radliya@yahoo.com

Nama Mahasiswa	
NIM	
Kelas	

Kompetensi Dasar

Memahami tahapan normalisasi data dalam melakukan perancangan basis data.

Pokok Bahasan

Normalisasi Data

1. Pengantar normalisasi
 - a. Definisi normalisasi
 - b. Jenis atribut
 - c. Domain dan tipe data
2. Anomali
 - a. Anomali penyisipan
 - b. Anomali pengubahan
 - c. Anomali penghapusan
3. Dependensi
 - a. Dependensi fungsional
 - b. Dependensi sepenuhnya
 - c. Dependensi parsial
 - d. Dependensi total
 - e. Dependensi transitif
4. Bentuk normal
 - a. Bentuk normal pertama
 - b. Bentuk normal kedua
 - c. Bentuk normal ketiga
 - d. Bentuk normal boyce-codd
 - e. Bentuk normal keempat
 - f. Bentuk normal kelima

I. Pengantar Normalisasi

1.1. Definisi Normalisasi

Perancangan basis data diperlukan agar tercipta basis data relasional yang efisien dalam penggunaan ruang penyimpanan, cepat dalam pengaksesan dan mudah dalam pemanipulasian (tambah, ubah, hapus) data. Dalam merancang basis data relasional, kita dapat melakukannya dengan cara:

1. Melakukan normalisasi data, lalu membuat model *Entity-Relationship*.

2. Membuat model *Entity-Relationship* terlebih dahulu, lalu melakukan normalisasi data.

Dalam model *Entity-Relationship* (E-R) kelompok-kelompok data dan relasi antarkelompok data tersebut diwujudkan/direpresentasikan dalam bentuk diagram. Normalisasi sendiri merupakan cara pendekatan lain yang tidak secara langsung berkaitan dengan model data, tetapi dengan menerapkan sejumlah aturan dan kriteria standar untuk menghasilkan struktur tabel yang normal.

Dalam pendekatan normalisasi, item-item data ditempatkan dalam baris dan kolom pada tabel-tabel relasional dengan sejumlah aturan tentang keterhubungan antara item-item data tersebut. Sementara pendekatan model E-R, lebih tepat dilakukan jika yang telah diketahui baru prinsip-prinsip sistem secara keseluruhan (belum diketahui item-item data yang digunakan pada sistem). Pada prakteknya, kedua pendekatan ini bisa dilakukan secara bergantian. Dari fakta yang telah kita miliki, kita lakukan normalisasi. Untuk kepentingan evaluasi dan dokumentasi, hasil normalisasi kita wujudkan dalam sebuah model data. Model data yang sudah jadi tersebut bisa saja dimodifikasi dengan pertimbangan tertentu. Hasil modifikasinya kemudian kita implementasikan dalam bentuk sejumlah struktur tabel dalam sebuah basis data. Struktur ini dapat kita uji kembali dengan menerapkan aturan normalisasi, hingga akhirnya kita peroleh sebuah struktur basis data yang benar-benar efektif dan efisien.

Menurut Kadir (2009 : 116) normalisasi adalah suatu proses yang digunakan untuk menentukan pengelompokan atribut-atribut dalam sebuah relasi/tabel sehingga diperoleh relasi yang berstruktur baik. Dalam hal ini yang dimaksud dengan berstruktur baik adalah relasi/tabel yang memenuhi kondisi sebagai berikut:

1. Mengandung redundansi sesedikit mungkin, dan
2. Memungkinkan baris-baris dalam relasi/tabel disisipkan, dimodifikasi, dan dihapus tanpa menimbulkan kesalahan atau ketidakkonsistenan.

1.2. Jenis Atribut

Atribut adalah suatu nama untuk kolom yang terdapat pada sebuah relasi. Atribut juga sering disebut sebagai kolom data atau *field*. Penerapan aturan-aturan normalisasi terhadap atribut-atribut pada sebuah tabel bisa berdampak pada penghilangan kolom tertentu, penambahan kolom baru, atau bahkan penambahan tabel baru. Atribut harus diberi nama yang unik dan tidak menggunakan spasi agar mudah pada saat

implementasi rancangan basis data. Atribut dapat dibedakan berdasarkan sejumlah pengelompokan (jenis atribut).

1.2.1. Atribut Kunci (*Key*) dan Atribut Deskriptif

Atribut kunci (*key*) adalah satu atau gabungan dari beberapa atribut yang dapat membedakan semua baris data dalam tabel secara unik. Ada beberapa macam *key* yang dapat diterapkan pada suatu tabel, yaitu:

1. *Superkey*
2. *Candidate Key*
3. *Primary Key*
4. *Foreign Key*

Keempat atribut kunci (*key*) tersebut sudah dijelaskan pada materi sebelumnya. Atribut ini dapat digunakan untuk tujuan identifikasi.

Atribut deskriptif adalah atribut-atribut yang bukan merupakan atribut *primary key* pada sebuah tabel. Atribut ini digunakan untuk tujuan informasi.

1.2.2. Atribut Sederhana (*Simple Attribute*) dan Atribut Komposit (*Composite Attribute*)

Atribut sederhana adalah atribut atomic yang tidak dapat dipilah lagi. Sedangkan atribut komposit merupakan atribut yang masih dapat diuraikan lagi menjadi sub-sub atribut yang masing-masing memiliki makna.

nim	nama_mhs	alamat_mhs
10507234	Alam Nurjaya	Jl. Dipatiukur No.91, Bandung, 40135
10507235	Bani Isro	Jl. Cijerah No.20, Cimahi 40533
10507236	Ningsih Amira	Jl. Raya Timur No.321, Tasikmalaya 46416

alamat	kota	kode_pos
Jl. Dipatiukur No.91	Bandung	40135
Jl. Cijerah No.20	Cimahi	40533
Jl. Raya Timur No.321	Tasikmalaya	46416

Gambar 1. Atribut Sederhana dan Atribut Komposit

Pada gambar 1 di atas, atribut *nim* dapat dikategorikan sebagai atribut sederhana, sedangkan atribut *alamat_mhs* dapat dikategorikan sebagai atribut komposit karena atribut tersebut dapat diuraikan menjadi beberapa subatribut seperti *alamat*, *kota* dan *kode_pos* yang masing-masing memiliki makna.

1.2.3. Atribut Bernilai Tunggal (*Single-Valued Attribute*) dan Atribut Bernilai Banyak (*Multivalued Attribute*)

Atribut bernilai tunggal ditujukan pada atribut-atribut yang memiliki hanya satu nilai untuk setiap baris data. Sedangkan atribut bernilai banyak ditujukan pada atribut-atribut yang dapat kita isi dengan lebih dari satu nilai, tetapi jenisnya sama.

Atribut Bernilai Tunggal			Atribut Bernilai Banyak
nim	nama_mhs	alamat_mhs	hobby
10507234	Alam Nurjaya	Jl. Dipatiukur No.91, Bandung, 40135	Futsal Berenang
10507235	Bani Isro	Jl. Cijerah, Cimahi 40533	Basket
10507236	Ningsih Amira	Jl. Raya Timur, Tasikmalaya 46416	Baca Buku Melukis

Gambar 2. Atribut Bernilai Tunggal dan Atribut Bernilai Banyak

Pada gambar 2 di atas atribut *nim*, *nama_mhs*, *alamat_mhs* merupakan atribut bernilai tunggal, sedangkan atribut *hobby* merupakan atribut bernilai banyak karena ada beberapa mahasiswa yang memiliki lebih dari satu hobby.

1.2.4. Atribut Harus Bernilai (*Mandatory Attribute*) dan Atribut Bernilai Null

Ada sejumlah atribut pada sebuah tabel yang kita tetapkan harus berisi data. Jadi nilainya tidak boleh kosong. Atribut semacam ini disebut *mandatory attribute* (atribut harus bernilai). Sedangkan kebalikan dari atribut tersebut adalah *non mandatory attribute* (atribut yang boleh tidak bernilai). Contohnya dapat dilihat pada gambar 3 di bawah ini.

Mandatory Attribute		No Mandatory Attribute	
nim	nama_mhs	alamat_mhs	hobby
10507234	Alam Nurjaya	Jl. Dipatiukur No.91, Bandung, 40135	Futsal Berenang
10507235	Bani Isro	Jl. Cijerah, Cimahi 40533	Basket
10507236	Ningsih Amira	Jl. Raya Timur, Tasikmalaya 46416	Baca Buku Melukis

Berisi Null, karena datanya belum siap
Berisi Null, karena memang tidak punya hobby

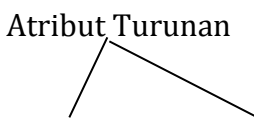
Gambar 3. *Mandatory Attribute* dan Nilai Null

Nilai *Null* tidak sama dengan spasi, walaupun pada waktu nilai ditampilkan sama-sama tidak memperlihatkan apa-apa. Perbedaan tersebut dapat ditinjau dari segi makna maupun dari segi representasi fisik. Dari segi makna, pengisian spasi ke suatu atribut berarti atribut tersebut memiliki nilai yaitu spasi, sedangkan pengisian nilai *Null* berarti atribut tersebut belum/tidak memiliki nilai. Dari segi representasi fisik, nilai spasi ekuivalen dengan karakter ke-32 dalam tabel ASCII, sedangkan nilai *Null* ekuivalen dengan karakter ke-0.

1.2.5. Atribut Turunan (*Derived Attribute*)

Atribut turunan adalah atribut yang nilai-nilainya diperoleh dari pengolahan atau dapat diturunkan dari atribut atau tabel lain yang berhubungan. Atribut tersebut sebetulnya dapat ditiadakan dari sebuah tabel, karena nilai-nilainya bergantung pada nilai yang ada di atribut lainnya.

Atribut Turunan



nim	nama_mhs	alamat_mhs	angkatan	ip
10507234	Alam Nurjaya	Jl. Dipatiukur No.91, Bandung, 40135	2007	3.64
10507235	Bani Isro	Jl. Cijerah, Cimahi 40533	2007	3.46
10507236	Ningsih Amira	Jl. Raya Timur, Tasikmalaya 46416	2007	3.87

Gambar 4. Atribut Turunan

Pada gambar 4 di atas nilai atribut *angkatan* dapat diketahui dari atribut *nim*, dimana karakter ke 4 dan 7 dari *nim* menyatakan dua digit akhir tahun masuknya mahasiswa yang bersangkutan. Sedangkan untuk atribut *ip* diperoleh dari pengolahan data yang melibatkan *indek_nilai* dari tabel *nilai* dan atribut *sks* yang ada di tabel *mata_kuliah*.

1.3. Domain dan Tipe Data

Domain adalah seluruh kemungkinan nilai yang dapat diberikan ke suatu atribut. Sebagai contoh, kemungkinan nilai untuk atribut *sks* adalah 1, 2, 3, 6. Sedangkan tipe data yang dapat digunakan untuk atribut tersebut adalah integer, meskipun integer memungkinkan kita menyimpan data angka yang bulat Antara -32,768 hingga 32,767. Dalam menentukan tipe data sebuah atribut sebaiknya terlebih dahulu kita melihat domain dari atribut tersebut.

II. Anomali

Normalisasi bertujuan untuk meminimalkan redundansi data karena redundansi data dapat menimbulkan masalah yang disebut anomali. Anomali adalah masalah yang timbul dalam relasi/tabel ketika terjadi pemutakhiran data di dalam relasi/tabel.

Terdapat 3 jenis anomali ada, diantaranya:

1. Anomali Penyisipan
2. Anomali Pengubahan
3. Anomali Penghapusan

Pada gambar 5 dan 6 di bawah ini berisi informasi yang sebenarnya sama, tetapi mempunyai efek berbeda yang terkait dengan anomali.

Tabel barang_pemasok

kode_barang	nama_barang	harga_jual	kode_pemasok	nama_pemasok	kota
T-001	TV ABC 14"	600000	P22	PT. Citra Jaya	Bogor
T-002	TV ABC 21"	950000	P22	PT. Citra Jaya	Bogor
T-003	TV XYZ 14"	450000	P11	PT. Amerta	Bandung
T-004	TV Rhino 29"	1750000	P33	PT. Kartika	Yogya
T-005	TV Kirana 14"	475000	P44	PT. Nindya	Tangerang

Gambar 5. Relasi/Tabel barang_pemasok

Tabel barang

kode_barang	nama_barang	harga_jual	kode_pemasok
T-001	TV ABC 14"	600000	P22
T-002	TV ABC 21"	950000	P22
T-003	TV XYZ 14"	450000	P11
T-004	TV Rhino 29"	1750000	P33
T-005	TV Kirana 14"	475000	P44

Tabel pemasok

kode_pemasok	nama_pemasok	kota
P11	PT. Amerta	Bandung
P22	PT. Citra Jaya	Bogor
P33	PT. Kartika	Yogya
P44	PT. Nindya	Tangerang

Gambar 6. Relasi/Tabel barang dan Relasi/Tabel pemasok

2.1. Anomali Penyisipan

Anomali penyisipan adalah masalah yang terjadi ketika suatu baris disisipkan ke dalam tabel. Anomali ini dapat muncul pada tabel barang_pemasok yang ada pada gambar 5. Contoh anomaly pada tabel barang_pemasok:

1. Apabila ada pemasok baru dengan nama PT. Santosa yang berlokasi di Bekasi dan kode pemasok P55. Data pemasok tersebut dapat dimasukkan apabila sudah ada barang yang dipasok.

2. Pemasok dengan kode P33 akan memasukan barang baru berupa TV Toslila 29” dan harga jual yang ditetapkan 2000000. Maka pada saat dimasukan, data pemasok dengan kode P11 (Nama pemasok dan lokasinya) perlu diisikan ulang. Masalahpun bertambah apabila data lokasi pemasok yang dimasukan adalah Yogyakarta (bukan Yogya) maka terjadi ketidakkonsistenan data lokasi untuk pemasok tersebut.

Permasalahan tersebut tidak mungkin terjadi pada tabel barang dan tabel pemasok yang tercantum di gambar 6. Karena apabila ada pemasok baru maka cukup dimasukan pada tabel pemasok, dan apabila ada barang baru dengan pemasok yang sudah ada maka cukup dimasukan ke dalam tabel barang.

2.2. Anomali Pengubahan

Anomali pengubahan adalah masalah yang timbul ketika data dalam tabel diubah. Contohnya apabila kita akan mengubah data lokasi pemasok untuk kode pemasok P22 pada tabel barang_pemasok yang ada di gambar 5. Lokasi pemasok dengan kode pemasok P22 berpindah dari Bogor ke Bekasi, sedangkan yang dirubah hanya pada baris pertama, sedangkan pada baris ke dua tidak dirubah. Hal tersebut dapat menimbulkan kerancuan atau ketidakkonsistenan. Sedangkan pada gambar 6, hal tersebut cukup dilakukan pada satu baris di tabel pemasok.

2.3. Anomali Penghapusan

Anomali penghapusan adalah masalah yang timbul ketika suatu baris dalam tabel dihapus. Pada saat sebuah baris dihapus terdapat data lain yang hilang. Sebagai contoh pada tabel barang_pemasok di gambar 5, apabila akan menghapus kode barang T-003 maka data pemasok dengan kode pemasok P11 akan ikut terhapus. Hal demikian tidak terjadi pada tabel barang yang ada di gambar 6, karena data pemasok tetap tersimpan pada tabel pemasok.

III. Dependensi

Dependensi menjelaskan hubungan antara atribut dengan atribut lainnya, atau secara lebih khusus menjelaskan nilai suatu atribut yang menentukan nilai atribut lainnya. Dependensi ini kelak menjadi acuan bagi pendekomposisi data kedalam bentuk yang paling efisien. Ada beberapa jenis dependensi yaitu diantaranya:

1. Dependensi Fungsional
2. Dependensi Sepenuhnya
3. Dependensi Parsial
4. Dependensi Total

5. Dependensi Transitif

3.1. Dependensi Fungsional

Dependensi fungsional adalah kekangan antara dua buah atribut atau dua buah himpunan. Suatu atribut Y mempunyai dependensi fungsional terhadap atribut X jika dan hanya jika setiap nilai X berhubungan dengan sebuah nilai Y. Dependensi fungsional Y terhadap X dapat dinotasikan sebagai berikut:

$X \rightarrow Y$

Notasi tersebut dapat dibaca dengan:

1. X panah Y
2. X menentukan Y
3. Y tergantung secara fungsional pada X

Contohnya pada tabel *barang_pemasok* (gambar 5) berlaku penotasian sebagai berikut:

$\text{kode_barang} \rightarrow \text{nama_barang}$

Dimana setiap *kode_barang* pasti akan berhubungan dengan hanya satu *nama_barang*. Misalnya, kode barang T-001 hanya berlaku untuk nama barang TV ABC 14".

Sebuah atribut juga bisa bergantung pada lebih satu atribut. Hal tersebut dapat dinotasikan sebagai berikut:

$\{X, Y\} \rightarrow Z$

Notasi di atas menyatakan bahwa atribut Z mempunyai dependensi fungsional terhadap pasangan atribut X dan Y. Sebagai contoh perhatikan tabel/relasi dalam gambar 7 di bawah ini.

Tabel dosen_pendidikan

no_dosen	nama_dosen	jenis_kelamin	pendidikan	tahun_lulus
D41	Rahayu Febrianti	Wanita	S1	1987
D41	Rahayu Febrianti	Wanita	S2	1990
D42	Amira Mari	Wanita	S1	1988
D42	Amira Mari	Wanita	S2	1990
D42	Amira Mari	Wanita	S3	1998
D43	Bara Adipura	Pria	S1	1994

Gambar 7. Relasi/Tabel *dosen_pendidikan*

Contohnya pada tabel *dosen_pendidikan* (gambar 7) berlaku penotasian sebagai berikut:

$\{\text{no_dosen}, \text{pendidikan}\} \rightarrow \text{tahun_lulus}$

Dimana tidak setiap *no_dosen* menentukan *tahun_lulus*, tetapi *tahun_lulus* ditentukan oleh perpaduan antara *no_dosen* dengan *pendidikan*. Misalnya, dosen dengan nomor D41 lulus S1 pada tahun 1987.

3.2. Dependensi Sepenuhnya

Suatu atribut Y dikatakan memiliki dependensi sepenuhnya terhadap X apabila memenuhi dua kondisi berikut:

1. Y mempunyai dependensi fungsional terhadap X,
2. Y **tidak** memiliki dependensi terhadap bagian dari X.

Contohnya pada tabel *dosen_pendidikan* (gambar 7) berlaku penotasian sebagai berikut:

$\{no_dosen, pendidikan\} \rightarrow tahun_lulus$

Dimana tidak setiap *no_dosen* menentukan *tahun_lulus*, dan tidak setiap *pendidikan* menentukan *tahun_lulus*. Atau dapat dilihat bahwa atribut *no_dosen* tidak berhubungan dengan satu nilai *tahun_lulus*, dan atribut *pendidikan* tidak berhubungan dengan satu nilai *tahun_lulus*.

3.3. Dependensi Parsial

Dependensi parsial merupakan kebalikan dari dependensi sepenuhnya. Dimana suatu atribut Y dikatakan memiliki dependensi parsial terhadap X apabila memenuhi dua kondisi sebagai berikut:

1. Y adalah atribut non-kunci utama dan X adalah kunci utama,
2. Y memiliki dependensi terhadap bagian dari X (tatapi tidak terhadap keseluruhan dari X)

Sebagai contoh pada tabel *dosen_pendidikan* (gambar 7) yang memiliki kunci utama berupa $\{no_dosen, pendidikan\}$. Atribut *jenis_kelamin* yang memiliki dependensi terhadap *no_dosen* (bagian dari kunci utama) merupakan dependensi parsial.

3.4. Dependensi Total

Suatu atribut Y dikatakan memiliki dependensi total terhadap X jika memenuhi dua kondisi sebagai berikut:

1. Y memiliki dependensi fungsional terhadap X
2. X memiliki dependensi fungsional terhadap Y

Contohnya pada tabel *pemasok* (gambar 6) berlaku penotasian sebagai berikut:

$kode_pemasok \rightarrow nama_pemasok$

$nama_pemasok \rightarrow kode_pemasok$

Dengan demikian berlaku penotasian sebagai berikut:

$kode_pemasok \leftrightarrow nama_pemasok$

Sebuah nilai *kode_pemasok* hanya akan berpasangan dengan sebuah *nama_pemasok*, dan begitu juga sebaliknya. Hal tersebut dengan asumsi tidak akan ada dua pemasok yang namanya sama.

3.5. Dependensi Transitif

Suatu atribut Z dikatakan memiliki dependensi transitif terhadap X apabila memenuhi dua kondisi sebagai berikut:

1. Z memiliki dependensi fungsional terhadap Y
2. Y memiliki dependensi fungsional terhadap X

Dependensi transitif dapat dinotasikan sebagai berikut:

$X \rightarrow Y \rightarrow Z$

Contohnya pada tabel *barang_pemasok* (gambar 5) berlaku penotasian sebagai berikut:

$kode_barang \rightarrow kode_pemasok \rightarrow nama_pemasok$

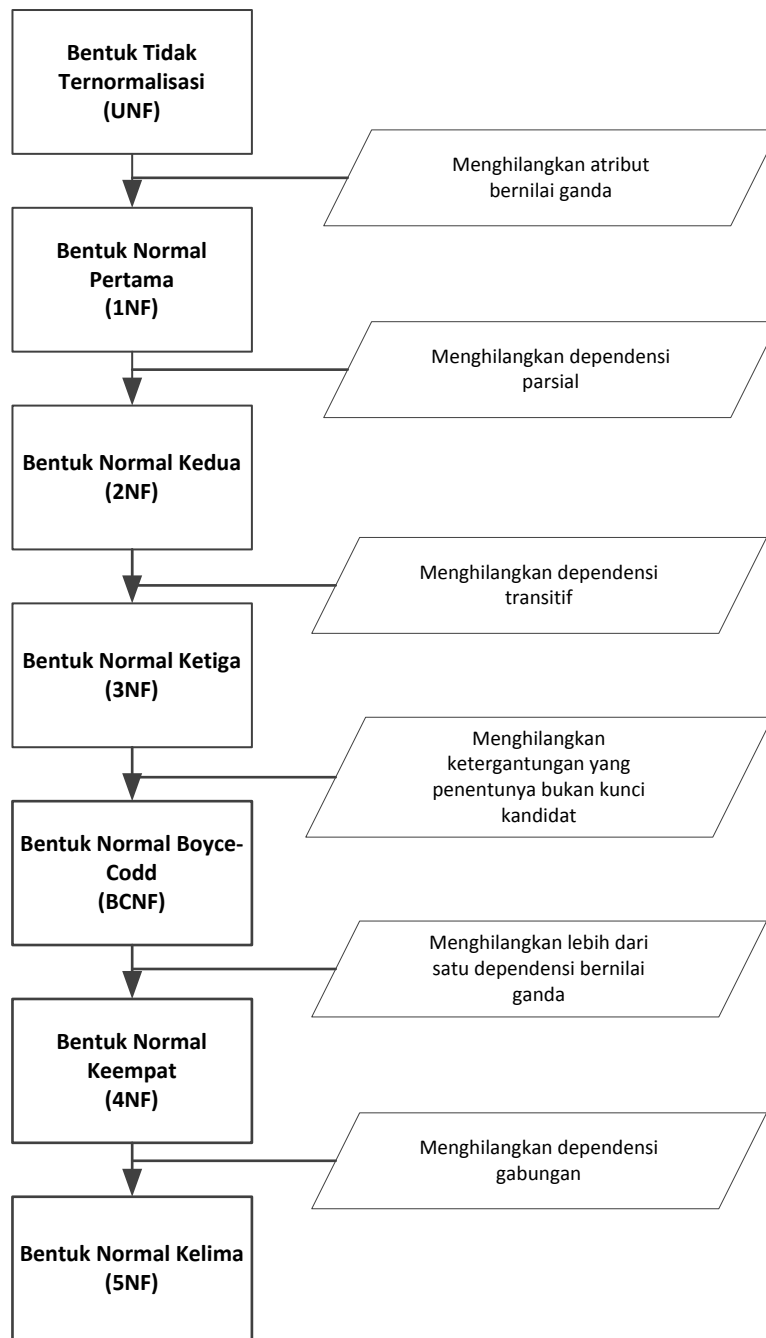
Dimana setiap *nama_pemasok* memiliki dependensi fungsional terhadap *kode_pemasok*, dan *kode_pemasok* memiliki dependensi fungsional terhadap *kode_barang*. Dengan demikian maka *nama_pemasok* memiliki dependensi transitif terhadap *kode_barang*.

IV. Bentuk Normal

Normalisasi sendiri dilakukan melalui sejumlah langkah. Setiap langkah berhubungan dengan bentuk normal (*normal form*) tertentu. Dalam hal ini yang disebut dengan bentuk normal adalah suatu keadaan relasi/tabel yang dihasilkan oleh penerapan aturan-aturan yang berhubungan dengan dependensi fungsional terhadap relasi/tabel tersebut. Aturan-aturan tersebut diterapkan pada beberapa bentuk normal sebagai berikut:

1. Bentuk Normal Pertama (1NF/ *First Normal Form*)
2. Bentuk Normal Kedua (2NF/ *Second Normal Form*)
3. Bentuk Normal Ketiga (3NF/ *Third Normal Form*)
4. Bentuk Normal Boyce-Codd (BCNF/ *Boyce-Codd Normal Form*)
5. Bentuk Normal Keempat (4NF/ *Fourth Normal Form*)
6. Bentuk Normal Kelima (5NF/ *Fifth Normal Form*)

Adapun hubungan keenam bentuk normal tersebut dapat dilihat pada gambar 8 di bawah ini.



Gambar 8. Langkah-Langkah Dalam Normalisasi

4.1. Bentuk Normal Pertama (1NF/ *First Normal Form*)

Suatu relasi/tabel dikatakan dalam bentuk normal pertama apabila setiap atribut bernilai tunggal (*Atomic Value*) untuk setiap barisnya. Untuk membentuk relasi/tabel agar berada dalam bentuk normal pertama, perlu langkah untuk menghilangkan atribut-atribut yang bernilai ganda.

Sebagai contoh, terdapat suatu bentuk yang belum ternormalisasi (UNF/*UnNormalized Form*) seperti yang ditunjukkan pada Gambar 9 (tabel tersebut menyatakan informasi tentang pegawai dank lien yang ditanganinya).

no_pegawai	nama_pegawai	no_klien	nama_klien
P27	Rahayu Febrianti	K01 K02 K04	Rini Suswandi Dani Damhudi Fatwa Sari
P28	Danang	K03 K07	Randa Irwanda Suci Jelita
P29	Amira Mari	K05	Febrianti
P30	Riki Maenaki	K06 K08	Siti Aminarti Sandi Sunardi

Gambar 9. Contoh Bentuk yang Belum Ternormalisasi

Untuk menghilangkan atribut yang bernilai ganda, bentuk sejumlah baris hingga setiap sel berisi satu nilai. Kemudian bagian yang kosong diisi dengan data yang sesuai. Pada gambar 10 di bawah ini menunjukkan hasil setelah semua sel diatur bernilai tunggal.

Tabel pegawai_klien

no_pegawai	nama_pegawai	no_klien	nama_klien
P27	Rahayu Febrianti	K01	Rini Suswandi
P27	Rahayu Febrianti	K02	Dani Damhudi
P27	Rahayu Febrianti	K04	Fatwa Sari
P28	Danang	K03	Randa Irwanda
P28	Danang	K07	Suci Jelita
P29	Amira Mari	K05	Febrianti
P30	Riki Maenaki	K06	Siti Aminarti
P30	Riki Maenaki	K08	Sandi Sunardi

Gambar 10. Bentuk Normal Pertama

Hal penting yang lainnya yang perlu dilakukan setelah melakukan normalisasi ke bentuk pertama adalah menentukan kunci utamanya. Kunci utama dapat dipilih melalui determinan-determinan (suatu penentu) yang muncul dalam relasi/tabel yang membuat setiap baris dapat diidentifikasi secara unik (tidak ada yang kembar). Kalau tidak ada determinan dengan satu atribut yang memenuhi, pilihlah gabungan atribut yang dapat digunakan untuk membedakan antara satu baris dengan baris lainnya. Sebagai contoh, dependensi fungsional yang ada pada relasi/tabel pegawai_klien yaitu:

no_pegawai -> nama_pegawai

no_klien -> nama_klien

Determinan *no_pegawai* atau *no_klien* tidak dapat digunakan sebagai kunci utama. Satu-satunya kunci utama yang bisa digunakan adalah $\{no_pegawai, no_klien\}$. Relasi/tabel yang berada pada bentuk normal pertama ada kemungkinan masih

mengandung anomali. Sebagai contoh, perhatikan gambar 10 yang masih terdapat beberapa anomali diantaranya:

1. **Anomali penyisipan.** Bila terdapat klien baru dan klien tersebut belum ditentukan akan ditangani oleh seseorang pegawai, penyisipan tidak dapat dilakukan karena kunci utama tidak boleh berupan *Null* (tidak diisi).
2. **Anomali pengubahan.** Bila nama pegawai diubah (karena ada kesalahan), ada kemungkinan lebih dari satu baris yang harus dimodifikasi. Apabila salah satu ada yang tidak diubah untuk nama yang sama, akan muncul ketidakkonsistenan.
3. **Anomali penghapusan.** Jika klien bernama Febrianti dihapus maka pegawai bernama Amira Mari akan ikut terhapus.

Maka dari itu relasi/tabel yang memenuhi bentuk normal pertama masih harus diproses untuk menjadi bentuk normal kedua.

4.2. Bentuk Normal Kedua (2NF/ *Second Normal Form*)

Suatu relasi/tabel dikatakan dalam bentuk normal kedua apabila:

1. Berada pada bentuk normal pertama.
2. Semua atribut bukan kunci memiliki dependensi sepenuhnya terhadap kunci utama atau tidak mengandung dependensi parsial.

Sebagai contoh pada gambar 10 (tabel pegawai_klien) masih terdapat dependensi parsial diantaranya:

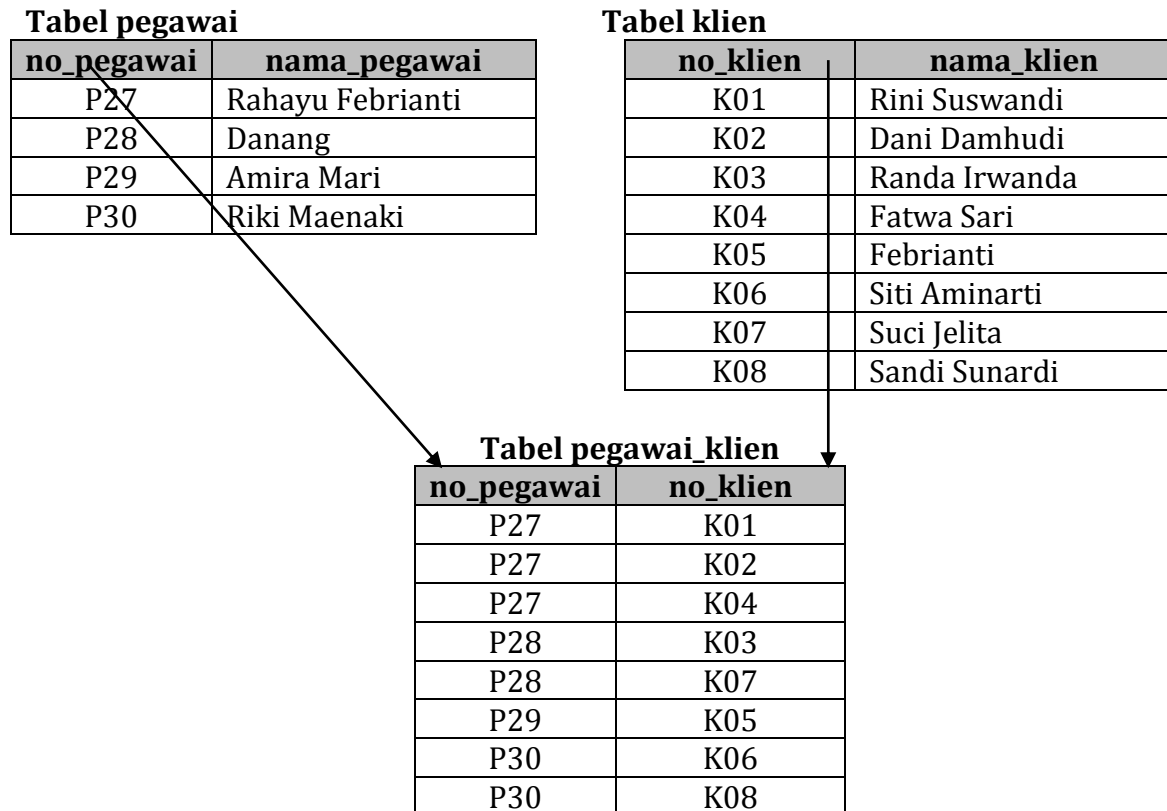
1. $\{no_pegawai, no_klien\} \rightarrow nama_pegawai$, dimana $nama_pegawai$ hanya bergantung pada $no_pegawai$ (bagian dari kunci utama).
2. $\{no_pegawai, no_klien\} \rightarrow nama_klien$, dimana $nama_klien$ hanya bergantung pada no_klien (bagian dari kunci utama).

Agar relasi/tabel yang mengandung dependensi parsial memenuhi bentuk normal kedua, dependensi parsialnya harus dihilangkan. Adapun cara untuk mengkonversi bentuk normal pertama ke bentuk normal kedua adalah sebagai berikut:

1. Ubahlah setiap dependensi parsial menjadi sebuah relasi/tabel, dengan kunci utama adalah determinannya.
2. Ubahlah dependensi yang terkait langsung dengan kunci utama sebagai relasi/tabel tersendiri dan kunci utamanya adalah kunci utama dalam relasi/tabel semula.

Berdasarkan cara diatas maka akan tercipta dua buah relasi/tabel baru karena terdapat dua buah dependensi parsial. Selain itu, langkah nomor dua juga menghasilkan

sebuah relasi/tabel. Dengan demikian relasi/tabel pegawai_klien didekomposisi (dipecah) menjadi tiga buah relasi/tabel seperti pada gambar 11 di bawah ini.



Gambar 11. Bentuk Normal Kedua

4.3. Bentuk Normal Ketiga (3NF/ *Third Normal Form*)

Suatu relasi/tabel dikatakan dalam bentuk normal ketiga apabila:

1. Berada pada bentuk normal kedua.
2. Setiap atribut bukan kunci tidak memiliki dependensi transitif terhadap kunci utama.

Agar suatu relasi/tabel masuk ke dalam bentuk normal ketiga, dependensi transitif harus dibuang. Adapun cara mendekomposisi relasi/tabel yang mengandung dependensi transitif adalah sebagai berikut:

1. Bentuk relasi/tabel yang mewakili dependensi fungsional yang tidak melibatkan kunci utama dalam relasi semula. Determinannya menjadi kunci utama relasi yang dibentuk.
2. Bentuk relasi/tabel yang berisi kunci utama relasi semula. Kemudian pindahkan semua atribut bukan kunci utama yang bergantung pada kunci utama tetapi tidak bergantung pada determinan lain ke relasi tersebut. Jadikan atribut yang menjadi kunci utama relasi semula sebagai kunci utama relasi baru. Adapun atribut yang

berasal dari determinan yang menjadi perantara dalam dependensi transitif akan bertindak sebagai kunci tamu/asing.

Sebagai contoh, pada tabel *barang_pemasok* (gambar 5) terdapat dependensi transitif sebagai berikut:

kode_barang -> kode_pemasok -> {nama_pemasok, kota}

Berdasarkan cara di atas, relasi/tabel *barang_pemasok* (gambar 5) dipecah menjadi tabel *barang* dan tabel *pemasok* seperti pada gambar 6.

4.4. Bentuk Normal Boyce-Codd (BCNF/ *Boyce-Codd Normal Form*)

Suatu relasi/tabel dikatakan dalam bentuk normal *Boyce-Codd* (BCNF) apabila setiap determinan (penentu) dalam suatu relasi/tabel berkedudukan sebagai kunci kandidat. Cara mengkonversi relasi yang telah memenuhi bentuk normal ketiga ke BCNF adalah:

1. Carilah semua penentu.
2. Bila terdapat penentu yang bukan berupa kunci kandidat, maka:
 - a. Pisahkan relasi tersebut.
 - b. Buat penentu tersebut sebagai kunci utama.

Sebagai contoh, misalkan terdapat relasi/tabel wawancara seperti pada gambar 12 di bawah ini.

Tabel wawancara

no_klien	tgl_wawancara	jam_wawancara	no_staff	ruangan
K33	2 Juli 2014	08.00	S44	R72
K34	2 Juli 2014	10.00	S44	R72
K35	2 Juli 2014	14.00	S45	R72
K36	3 Juli 2014	08.00	S44	R73
K37	3 Juli 2014	10.00	S45	R73

Gambar 12. Relasi/Tabel wawancara

Relasi/tabel tersebut jelas memenuhi bentuk normal ketiga karena tidak mengandung dependensi transitif. Terlebih dahulu kita lakukan pengkajian terhadap seluruh aturan yang bisa berlaku pada relasi tersebut:

1. Kunci utama pada relasi tersebut adalah {*no_klien, tgl_wawancara*}
2. Pada satu hari tertentu setiap staff diberi satu ruangan tertentu. Misalnya, pada tanggal 2 Juli 2014, staff S44 mendapat jatah ruangan R72.
3. Pada hari yang sama setiap ruang bisa dialokasikan untuk lebih dari satu staff. Sebagai contoh, R72 digunakan untuk S44 dan S45 (pada jam yang berlainan) pada tanggal 22 Juli 2014.

4. Setiap klien bisa diwawancarai lebih dari satu kali, tetapi hanya sekali dalam satu hari. Itulah sebabnya, *tgl_wawancara* perlu dijadikan bagian dari kunci utama, tetapi *jam_wawancara* tidak.

Anomali masih muncul pada relasi tersebut. Sebagai contoh, jika pada tanggal 2 Juli 2014 wawancara yang ditangani staff S44 dipindahkan ke ruangan R77 maka terdapat dua buah baris yang harus diperbaharui, kalau hanya satu maka akan timbul ketidakkonsistenan.

Agar relasi yang memenuhi bentuk normal ketiga berada pada bentuk BCNF, dependensi yang melibatkan atribut bukan kunci kandidat harus dinyatakan dalam relasi tersendiri dan atribut yang berkedudukan sebagai dependen dikeluarkan dari relasi semula. Dengan demikian relasi wawancara perlu didekomposisi menjadi seperti pada gambar 13.

Tabel jadwal_wawancara

no_klien	tgl_wawancara	jam_wawancara	no_staff
K33	2 Juli 2014	08.00	S44
K34	2 Juli 2014	10.00	S44
K35	2 Juli 2014	14.00	S45
K36	3 Juli 2014	08.00	S44
K37	3 Juli 2014	10.00	S45

Tabel staff_ruang

no_staff	tgl_wawancara	ruangan
S44	2 Juli 2014	R72
S45	2 Juli 2014	R72
S44	3 Juli 2014	R73
S45	3 Juli 2014	R73

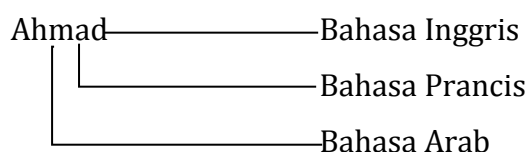
Gambar 13. Bentuk Normal *Boyce-Codd*

4.5. Bentuk Normal Keempat (4NF/ *Fourth Normal Form*)

Suatu relasi/tabel memenuhi bentuk normal ke empat (4NF) apabila:

1. Telah berada pada BCNF.
2. Tidak mengandung dependensi bernilai banyak (MVD/*Multi-Valued Dependency*).

MVD merupakan dependensi antara dua atribut dalam sebuah relasi dengan sifat untuk setiap nilai A terdapat sejumlah nilai B. Jadi sebuah nilai A berpasangan dengan sejumlah nilai B. Contohnya ditunjukkan pada gambar 14 di bawah ini.



Gambar 14. Gambaran Dependensi Bernilai Banyak

Contoh di atas menyatakan bahwa seseorang bisa memiliki kemampuan berbahasa asing lebih dari satu. Hubungan tersebut berada dalam sebuah relasi/tabel. Dependensi seperti itu digambarkan sebagai berikut:

nama ->> bahasa_asing

Tabel mdb

matakuliah	dosen	buku_wajib
Basis Data	Amri Yahya	Database Systems
Basis Data	Amri Yahya	Modern Database Management
Basis Data	Rini Subono	Database Systems
Basis Data	Rini Subono	Modern Database Management
Teknologi Informasi	Sunaryo Hadi	Information Technology Management
Teknologi Informasi	Sunaryo Hadi	Pengantar Teknologi Informasi
Teknologi Informasi	Karyo Junaedi	Information Technology Management
Teknologi Informasi	Karyo Junaedi	Pengantar Teknologi Informasi

Gambar 15. Contoh Relasi/Tabel dengan Dependensi Bernilai Banyak

Relasi mdb memiliki dua buah dependensi bernilai banyak, yaitu:

matakuliah ->> dosen

matakuliah ->> buku_wajib

Yang perlu diperhatikan adalah bahwa baik *dosen* maupun *buku_wajib* bersifat independen atau tidak saling berketergantungan. Keadaan seperti ini biasa dinyatakan seperti berikut:

matakuliah ->> dosen | buku_wajib

Relasi mdb memiliki dependensi bernilai banyak, tetapi sudah memenuhi bentuk normasi *Boyce-Codd*. Relasi tersebut memiliki redundansi data yang berlebihan. Suatu relasi yang mengandung dependensi bernilai banyak dapat dikonversi agar memenuhi bentuk normak keempat dengan menggunakan teorema Fagin. Teorema tersebut berbunyi sebagai berikut:

“Bila $R(A,B,C)$ merupakan suatu relasi, dengan A, B, dan C adalah atribut-atributnya, maka R dapat dipecah menjadi (A,B) dan (A,C) jika R memenuhi $MVD A \rightarrow B | C$ ”

Berdasarkan teorema tersebut, relasi mdb dapat dipecah menjadi seperti pada gambar 16 di bawah ini.

Tabel mdb

matakuliah	dosen
Basis Data	Amri Yahya
Basis Data	Rini Subono
Teknologi Informasi	Sunaryo Hadi
Teknologi Informasi	Karyo Junaedi

Tabel mdb

matakuliah	buku_wajib
Basis Data	Database Systems
Basis Data	Modern Database Management
Teknologi Informasi	Information Technology Management
Teknologi Informasi	Pengantar Teknologi Informasi

Gambar 16. Bentuk Normal Keempat

4.6. Bentuk Normal Kelima (5NF/ *Fifth Normal Form*)

Secara praktis, bentuk normal kelima adalah suatu keadaan yang membuat relasi/tabel yang telah memenuhi bentuk normal keempat tidak dapat didekomposisi menjadi relasi-relasi yang lebih kecil dengan kunci kandidat relasi-relasi pecahannya tersebut tidak sama dengan kunci kandidat relasi.

Sebagai contoh, perhatikan gambar 17 di bawah ini.

Tabel mahasiswa

nim	nama_mhs	jenis_kelamin	tanggal_lahir
10507234	Alam Nurjaya	Pria	2 November 1987
10507235	Bani Isro	Pria	2 Desember 1989
10507236	Ningsih Amira	Wanita	1 Februari 1989

Gambar 17. Bentuk Normal Kelima

Relasi tersebut dapat dipecah menjadi dua atau tiga relasi, yang apabila derekonstruksi akan membentuk kembali data semula. Relasinya bisa berupa seperti berikut:

R1 (nip*, nama_mhs)

R2 (nip*, jenis_kelamin)

R3 (nip*, tanggal_lahir)

Perhatikan bahwa kunci utama hasil dekomposisi R1, R2 dan R3 sama dengan kunci utama relasi mahasiswa. Hal ini menyatakan bahwa relasi mahasiswa sebenarnya telah memenuhi bentuk normal kelima.

V. Soal Latihan

1. Apa yang dimaksud dengan normalisasi?
2. Jelaskan mengenai jenis-jenis atribut?
3. Jelaskan perbedaan dan keterkaitan antara domain dengan tipe data?
4. Jelaskan mengenai anomali?
5. Jelaskan mengenai macam-macam dependensi?
6. Jelaskan langkah-langkah untuk memenuhi setiap bentuk normal?

VI. Materi Berikutnya

Pokok Bahasan	Praktikum 1: Normalisasi Data
Sub Pokok Bahasan	1. Analisis sistem informasi 2. Normalisasi data

VII. Daftar Pustaka

Fathansyah. 2012. Basis Data. Bandung: Informatika.

Kadir, A. 2009. Dasar Perancangan dan Implementasi Database Relasional. Yogyakarta: Andi.

Simarmata, J. 2007. Perancangan Basis Data. Yogyakarta: Andi.