BASIS DATA (BS203)

NORMALISASI

k_doroedi@yahoo.com
fb: NDoro Edi

Outline

- Latar belakang
- Anomali dan jenisnya
- Dependensi dan jenisnya
- Dekomposisi
- Bentuk Normal 1 (1NF)
- Bentuk Normal 2 (2NF)
- Bentuk Normal 3 (3NF)
- Bentuk Normal Boyce-Codd (BCNF)
- Bentuk Normal 4 (4NF)
- Bentuk Normal 5 (5NF)

Mengapa kita perlu Normalisasi?

- Ketika merancang basisdata menggunakan model relasional, kita sering menemukan beberapa alternatif pendefinisian himpunan skema relasi/ tabel
- Kita harus berhati-hati dalam memilih atribut – atribut apa saja yang dapat masuk ke dalam suatu skema relasi/ tabel

Mengapa kita perlu Normalisasi? (2)

- Perancangan basisdata melalui proses normalisasi memiliki keuntungan-keuntungan sbb:
  - Meminimalkan ukuran penyimpanan yang diperlukan untuk menyimpan data
  - Meminimalkan resiko inkonsistensi data pada basisdata
  - Meminimalkan anomali pada saat update data
  - Memaksimalkan struktur basisdata

Mengapa kita perlu Normalisasi? (3)

- Kegunaan:
  - Dipakai sebagai metodologi untuk menciptakan struktur tabel dalam basis data
  - Dipakai sebagai perangkat verifikasi terhadap tabel-tabel yang dihasilkan oleh metodologi lain (misalnya E-R)

Definisi Normalisasi

- Normalisasi adalah teknik desain yang secara luas digunakan sebagai panduan dalam merancang basisdata relasional
- Pada dasarnya, normalisasi merupakan suatu proses dengan dua langkah yaitu menyimpan data dalam bentuk tabular dengan cara menghapus kelompok data yang berulang dan kemudian menghapus duplikasi data dari tabel
- Dengan kata lain, kita bisa mengatakan bahwa Normalisasi → Proses menghilangkan redundansi data
- **Definisi**: proses untuk mengubah suatu relasi yang memiliki masalah tertentu ke dalam dua buah relasi atau lebih yang tidak memiliki masalah tersebut. Masalah yang dimaksud disebut juga anomali.

Masalah Redundansi

- Secara teoritis, kita dapat menyimpan semua atribut dalam suatu relation/ tabel. Artinya satu tabel mendeskripsikan seluruh sistem.
- Hanya saja, hal ini akan menimbulkan redundansi data.
- Contoh kasus: tabel mahasiswa

```
CREATE TABLE mahasiswa (
    NRP CHAR(10),
    Nama CHAR(30),
    Alamat CHAR(50),
    Hobby CHAR(20),
    PRIMARY KEY (NRP, Hobby)
)
```

Kasus Tabel Mahasiswa

NRP	Nama	Alamat	Hobby
1001	Nina	Jl. Siliwangi	Memancing
1001	Nina	Jl. Siliwangi	Menembak
1002	Kartini	Jl. Dago	Menembak
1002	Kartini	Jl. Dago	Berenang
1003	Nurul	Jl. Sabang	Berburu
1003	Nurul	Jl. Sabang	Berenang
1004	Coki	Jl. Pasteur	Memasak
1005	Wiwi	Jl. Tol	Menari

Masalah pada tabel Mahasiswa

- Rancangan tabel mahasiswa tsb sangat memungkinkan untuk terjadi redundansi data seperti yang terlihat pada tabel.
- Redundansi ini dapat menyebabkan masalah pada saat pembaruan data, atau yang biasa dikenal dengan Anomali (anomali pembaruan).
- Anomali terdiri dari 3 jenis:
  - Anomali update (peremajaan/ pembaruan)
  - Anomali insert (penyisipan/ penambahan)
  - Anomali delete (penghapusan)

1. Anomali Update

- Terjadi bila ada pengubahan pada sejumlah data yang mubazir, tetapi tidak seluruhnya diubah.
- Misalnya saja, tuple pertama dan kedua sbb:

1001	Nina	Jl. Siliwangi	Memancing
1001	Nina	Jl. Siliwangi	Menembak

- Lalu terjadi update:

  Nina berpindah ke alamat Jl. Mekarsari.

- Maka proses update memerlukan penbubahan alamat di dua tuple. Cara ini dinilai tidak wajar dan merepotkan.

2. Anomali Insert

- Terjadi jika pada saat penambahan hendak dilakukan ternyata ada elemen data yang masih kosong dan elemen data tersebut justru menjadi kunci.
- Misalnya, saat ingin menambah data baru ke tabel mahasiswa:

  NRP : 1007
  Nama : Budi
  Alamat : Jl. Budisari
  Hobby : ? (belum tahu)

- Tuple yang akan dibentuk: (1007, Budi, Jl. Budisari, NULL)
- Kolom Hobby merupakan bagian dari Primary Key, oleh karena itu tidak boleh bernilai NULL. Hal ini melanggar aturan Primary Key sehingga proses insert tidak akan berhasil

3. Anomali Delete

- Terjadi sekiranya suatu baris/ tuple yang tidak terpakai dihapus dan sebagai akibatnya terdapat data lain yang hilang.

1004	Coki	Jl. Pasteur	Memasak

- Misalnya, kita ingin menghapus Hobby “Memasak” untuk “Coki”
- Maka hal ini menyebabkan juga hilangnya data NRP, Nama, dan Alamat “Coki” dari basisdata. Padahal hal ini tidak dikehendaki.
- Kita juga tidak dapat mengganti “Memasak” dengan nilai NULL karena hal ini melanggar aturan PRIMARY KEY.

So What?

- Untuk menghilangkan Anomali-anomali tersebut, kita perlu men-dekomposisi yaitu memecah relation/tabel menjadi 2 tabel, yaitu: “Tabel Mahasiswa” dan “ Tabel Hobby”

NRP	Nama	Alamat
1001	Nina	Jl. Siliwangi
1002	Kartini	Jl. Dago
1003	Nurul	Jl. Sabang
1004	Coki	Jl. Pasteur
1005	Wiwi	Jl. Tol

NRP	Hobby
1001	Memancing
1001	Menembak
1002	Menembak
1002	Berenang
1003	Berburu
1003	Berenang
1004	Memasak
1005	Menari

Dependencies (Dependensi/Ketergantungan)

- Dependensi merupakan konsep yang mendasari normalisasi.
- Dependensi menjelaskan hubungan antar atribut, atau secara lebih khusus menjelaskan nilai suatu atribut yang menentukan nilai atribut lainnya.
- Dependensi ini kelak menjadi acuan bagi pendekomposisian data ke dalam bentuk yang paling efisien.
- Dua hal penting yang digunakan untuk mendefinisikan bentuk-bentuk normal:
  - Kebergantungan(hubungan) di antara atribut-atribut relasi
  - Kunci relasi
    Kunci relasi adalah himpunan atribut yang nilai-nilainya dapat mengidentifikasi baris-baris unik di relasi.

Sintaks FD

- Apabila **r** adalah suatu tabel/relation, kemudian **X** dan **Y** adalah atribut dari r. Maka kita bisa menyebutkan bahwa Y **secara fungsional bergantung pada X**, dengan menggunakan simbol berikut:

X → Y

(baca: "X functionally determines Y"atau"X panah Y")

Contoh FD

- Fakta bahwa seorang pegawai dengan IDPegawai mempunyai tglLahir dapat direpresentasikan dengan kebergantungan fungsional sbb:

IDPegawai → tglLahir

- Sisi kiri disebut determinan(penentu)
- Nilai di determinan dapat menentukan hanya satu nilai sisi kanan
- Jadi, satu nilai IDPegawai menentukan hanya satu nilai tglLahir
- Jika terdapat lebih dari satu nilai atribut di sisi kanan dapat diasosiasikan dengan satu nilai di sisi kiri, berarti tidak terdapat kebergantungan fungsional.

Contoh Kasus

- Misalnya, kita memiliki tabel sebagai berikut:

```
CREATE TABLE Shipments
(S# VARCHAR (5),
 CITY VARCHAR(10),
 P# VARCHAR (5),
 QTY NUMERIC(9),
 PRIMARY KEY (S#,P#),
 FOREIGN KEY (S#) REFERENCES S,
 FOREIGN KEY (P#) REFERENCES P
)
```

Tabel SHIPMENT

Shipment (S)	CITY	Product (P)	QTY
S1	London	P1	100
S1	London	P2	100
S2	Paris	P1	200
S2	Paris	P2	200
S3	Paris	P2	300
S4	London	P2	400
S4	London	P4	400
S4	London	P5	400

Contoh Kasus FD

- Pada tabel SHIPMENTS, terdapat hubungan FD sebagai berikut:

  { S# } → { CITY }

  Karena pada setiap tuple dari relation Shipments, setiap nilai S# memiliki CITY yang sama, yaitu:

  S1 → London
  S2 → Paris
  S3 → Paris
  S4 → London

S	CITY	P	QTY
S1	London	P1	100
S1	London	P2	100
S2	Paris	P1	200
S2	Paris	P2	200
S3	Paris	P2	300
S4	London	P2	400
S4	London	P4	400
S4	London	P5	400

Contoh Kasus FD (2)

Selain itu, terdapat juga beberapa FD yang lain, yaitu:

- „{ S#, P# } → { QTY }
- „{ S#, P# } → { CITY }
- „{ S#, P# } → { CITY, QTY }
- „{ S#, P# } → { S# }
- „{ S#, P# } → { S#, P#, CITY, QTY }
- „{ QTY } → { S# }

S	CITY	P	QTY
S1	London	P1	100
S1	London	P2	100
S2	Paris	P1	200
S2	Paris	P2	200
S3	Paris	P2	300
S4	London	P2	400
S4	London	P4	400
S4	London	P5	400

Jenis-jenis Kebergantungan Fungsional

1. Kebergantungan Fungsional (biasa)
2. Saling Bergantung (Kebergantungan Total)
3. Kebergantungan pada lebih dari satu atribut
4. Kebergantungan fungsional penuh (full functional dependency – FFD)
5. Kebergantungan Transitif

1. Kebergantungan Fungsional Biasa

- Suatu atribut X mempunyai dependensi fungsional terhadap atribut Y jika dan hanya jika setiap nilai X berhubungan dengan sebuah nilai Y.
- **X → Y** : “X secara fungsional menentukan Y”

2. Saling Bergantung (Kebergantungan Total)

- Suatu atribut Y mempunyai dependensi total terhadap atribut X (X dan Y saling bergantung) jika Y memiliki dependensi fungsional terhadap X dan X mempunyai dependensi fungsional terhadap Y.

Contoh:

- Jika suatu proyek mempunyai satu manajer dan masing-masing manajer hanya mengelola satu proyek, maka pernyataan kebergantungan fungsional yang ada adalah sebagai berikut:

  IDManajer → IDProyek

  IDProyek → IDManajer

  atau kita singkat sbb:

  IDManajer ↔ IDProyek

3. Kebergantungan pada lebih dari satu atribut

IDPegawai	IDProyek	TotalWaktuKeterlibatan
P001	Pj001	20
P003	Pj001	16
P002	Pj002	35
P002	Pj003	42
P003	Pj002	17
P003	Pj001	83
P004	Pj003	41

Waktu keterlibatan pegawai di suatu proyek adalah fakta mengenai asosiasi antara pegawai dan proyek. Nilai IDPegawai tidak cukup untuk memperoleh nilai tunggal TotalWaktuKeterlibatan karena pegawai dapat bekerja di lebih dari satu proyek. Nilai Total WaktuKeterlibatan akan berbeda untuk tiap proyek yang diikuti pegawai itu.

Kebergantungan ini ditunjukan dengan:

IDPegawai, IDProyek → TotalWaktuKeterlibatan

4. Kebergantungan Fungsional Penuh (Fully Functional Dependency - FFD)

- „Suatu atribut Y mempunyai dependensi fungsional penuh terhadap atribut X jika Y mempunyai dependensi fungsional terhadap X dan Y tidak memiliki dependensi terhadap bagian dari X.
- „Kebergantungan fungsional penuh adalah kebergantungan fungsional di mana tidak ada atribut-atribut yang tidak perlu yang berada di sisi determinan (sisi kiri).
- „Contoh:

  Kebergantungan fungsional berikut:

  (1) IDPegawai → tglLahir

  maka benar juga menyatakan:

  (2) IDPegawai, namaPegawai → tglLahir
- „Pada (2) sebenarnya namaPegawai tidak diperlukan untuk memperoleh tglLahir. IDPegawai sudah mencukupi untuk memperoleh nilai tglLahir. Jadi:
- „(1) merupakan kebergantungan fungsional penuh
- „(2) bukan kebergantungan fungsional penuh

5. Kebergantungan Transitif

- Atribut Z mempunyai dependensi transitif terhadap X bila Y memiliki dependensi fungsional terhadap X dan Z memiliki dependensi fungsional terhadap Y.
- „Contoh:

NRP	NAMA	ALAMAT	NO_HP
0372001	Shinta	Jl. Citarum 2	+62812382828
0371001	Rama	Jl. Macan 3	+62812382926
0472002	Shierly	Jl. Duku 5	+62812352429
0573001	Edo	Jl. Sabang 4	+62812382020

- NRP → NAMA
- „NAMA → NO_HP
- „NRP → NAMA → NO_HP
- „Asumsi: Nama unik

Proses Dekomposisi

- Tujuan perancangan basisdata adalah membangun relation-relation/tabel-tabel dengan redundansi minimal.
- Tabel seharusnya berbentuk normal setinggi mungkin.
- Pengkonversian satu bentuk normal ke bentuk normal yang lebih tinggi mengeliminasi satu jenis redundansi.
- Konversi ini dilakukan dengan mendekomposisi satu skema tabel menjadi sekumpulan skema tabel yang masing-masingnya memiliki bentuk normal yang lebih tinggi.

Sifat-sifat Dekomposisi

1. Dekomposisi tidak mengakibatkan munculnya atribut-atribut baru dan tidak mengakibatkan penghilangan atribut-atribut pada skema asal.
2. Dekomposisi tidak mengakibatkan munculnya kebergantungan fungsional baru, tapi boleh membuang.

Dekomposisi Tak Hilang

- „Dekomposisi ialah proses pemecahan sebuah relasi menjadi dua relasi atau lebih.
- „Dekomposisi tak hilang artinya bahwa tidak ada informasi yang hilang ketika relasi dipecah menjadi relasi-relasi lain.
- „Contoh:

NRP	NAMA	JURUSAN
95001	ALI	EKONOMI
95002	EDI	EKONOMI
95003	ALI	FISIKA

Contoh Dekomposisi

NRP	NAMA	JURUSAN
95001	ALI	EKONOMI
95002	EDI	EKONOMI
95003	ALI	FISIKA

Contoh Dekomposisi Tak Hilang:

NRP	NAMA
95001	ALI
95002	EDI
95003	ALI

NRP	JURUSAN
95001	EKONOMI
95002	EKONOMI
95003	FISIKA

Contoh Dekomposisi Hilang:

NRP	NAMA
95001	ALI
95002	EDI
95003	ALI

NAMA	JURUSAN
ALI	EKONOMI
EDI	EKONOMI
ALI	FISIKA

Bentuk Normal

- Teori Normalisasi berasal dari konsep normal forms(bentuk normal). Normal forms ini menyatakan tingkat redundansi yang terjadi pada suatu tabel.
- Suatu tabel relasional dikatakan berada dalam bentuk normal apabila memenuhi constraint tertentu.
- Ada 6 bentuk normal yang telah didefinisikan:
  - **Bentuk normal pertama (1NF) → umum**
  - **Bentuk normal kedua (2NF) → umum**
  - **Bentuk normal ketiga (3NF) → umum**
  - **Bentuk normal Boyce-Codd (BCNF) → revisi 3NF**
  - **Bentuk normal keempat (4NF) → kasus khusus (multivalue atribute)**
  - **Bentuk normal kelima (5NF) → kasus khusus (join table)**
- Jadi bisa dikatakan bahwa, normalisasi adalah pemrosesan tabel-tabel menjadi bentuk normal yang lebih tinggi.
- Dengan demikian, tujuan proses normalisasi adalah mengkonversi relation/tabel menjadi bentuk normal yang lebih tinggi.

Bentuk Normal (2): History

- Codd mendefinisikan bentuk normal pertama, kedua, dan ketiga pada makalahnya di tahun 1970.
- Bentuk normal ketiga kemudian diperbaiki sehingga mempunyai bentuk normal yang lebih baik yaitu BCNF(Boyce-Codd) pada tahun 1974.
- Fagin memperkenalkan bentuk normal keempat pada tahun 1977.
- Pada tahun 1979, Fagin kemudian memperkenalkan bentuk normal kelima.

Kriteria Proses Normalisasi

- Kriteria dalam proses normalisasi adalah
  - Kebergantungan fungsional (functional dependency) →1NF s/d BCNF
  - Kebergantungan banyak nilai(multivalue dependency) → 4NF
  - Kebergantungan join (join dependency)→ 5NF
- Ketiga tipe kebergantungan tersebut digunakan untuk menilai tabel-tabel yang dihasilkan dari konversi diagram ER.
- Proses normalisasi membentuk tabel-tabel bentuk normal menggunakan proses dekomposisi yaitu memecah satu tabel menjadi tabel-tabel berbentuk normal.
- Biasanya bentuk normal BCNF sudah memadai untuk berbagai aplikasi basisdata.

Bentuk Normal Pertama (First Normal Form – 1NF)

- Bentuk normal pertama adalah ekivalen dengan definisi model relasional.
- Relation/tabel adalah berbentuk normal pertama jika semua nilai atributnya adalah sederhana (bukan komposit)
- Dengan kata lain, suatu relasi dikatakan dalam bentuk normal pertama jika dan hanya jika setiap atribut bernilai tunggal untuk setiap baris (tidak memiliki atribut yang berulang).
- Diubah ke dalam bentuk normal dengan cara membuat setiap baris berisi kolom dengan jumlah yang sama dan setiap kolom hanya mengandung satu nilai.

Contoh A yang BUKAN 1NF

Tabel Pesanan

IDPesanan	tglPesanan	isiPesanan	
T20	6 Jul 2006	IDItem	Kuantitas
		PC6	24
		BW3	83
		Ty6	37
T33	6 Aug 2006	IDItem	Kuantitas
		PC5	800
		BW3	80
		Ty6	1000

Hasil Normalisasi Tahap 1
Tabel Pesanan

- „Tabel Pesanan tidak dalam bentuk 1NF karena nilai atribut isiPesanan bukan nilai sederhana, alias komposit.
- „Tabel tersebut dapat menjadi 1NF dengan membuat relasi tersebut menjadi sbb:

IDPesanan	TglPesanan	IDItem	Kuantitas
T20	6 Jul 2006	PC6	24
T20	6 Jul 2006	BW3	83
T20	6 Jul 2006	TY6	37
T33	6 Aug 2006	PC5	800
T33	6 Aug 2006	BW3	80
T33	6 Aug 2006	Ty6	1000

Contoh B Untuk 1NF

NIP	NAMA	JABATAN	KEAHLIAN	LAMA
107	ILHAM	ANALIS SENIOR	COBOL ORACLE	6 1
109	RIAN	ANALISIS YUNIOR	COBOL DBASE III+	2 2
112	FISIKA	PEMROGRAM	COBOL DBASE III+ SYBASE	1 1 1

Hasil Contoh B Untuk 1NF

NIP	NAMA	JABATAN	KEAHLIAN	LAMA
107	ILHAM	ANALIS SENIOR	COBOL	6
107	ILHAM	ANALIS SENIOR	ORACLE	1
109	RIAN	ANALISIS YUNIOR	COBOL	2
109	RIAN	ANALISIS YUNIOR	DBASE III+	2
112	FISIKA	PEMROGRAM	COBOL	1
112	FISIKA	PEMROGRAM	DBASE III+	1
112	FISIKA	PEMROGRAM	SYBASE	1

Contoh C Untuk 1NF

No	Tgl	I_1	I_2	I_3	I_4	Total (Rp)
50001	12/05/1997	P1	P2	P3	P4	45.000
50002	12/05/1997	P3	P5	P6		32.500
50003	13/05/1997	P1	P2			12.000

Hasil Contoh C Untuk 1NF

No_Pesanan	Tgl_Pesanan	Item	Total
50001	12/05/1997	P1	45.000
50001	12/05/1997	P2	45.000
50001	12/05/1997	P3	45.000
50001	12/05/1997	P4	45.000
50002	12/05/1997	P3	32.500
50002	12/05/1997	P5	32.500
50002	12/05/1997	P6	32.500
50003	13/05/1997	P1	12.000
50003	13/05/1997	P2	12.000

Kesimpulan 1NF

- Relasi yang memenuhi bentuk normal pertama umumnya memiliki masalah kemubaziran yang mengakibatkan ketidakkonsistenan pada saat pengubahan data.
- Dan walaupun ketidakkonsistenan dapat dihindari, terjadi ketidakefisienan sewaktu mengubah data.

Bentuk Normal Kedua (2NF)

- „Suatu relasi berada dalam bentuk normal kedua jika dan hanya jika:
  - berada pada bentuk normal pertama, dan
  - semua atribut bukan kunci memiliki dependensi sepenuhnya terhadap kunci primer. Dengan kata lain, setiap atribut harus bergantung pada kunci primer.

Bentuk Normal Kedua (2NF)

- Diubah ke 2NF dengan melakukan dekomposisi terhadap relasi tersebut.
- Dapat dilakukan dengan menggambarkan diagram dependensi fungsional terlebih dahulu.
- Berdasarkan diagram ini, relasi dalam bentuk 1NF dapat dipecah ke dalam sejumlah relasi.

Contoh A Untuk 2NF

NIP	NAMA	JABATAN	KEAHLIAN	LAMA
107	ILHAM	ANALIS SENIOR	COBOL	6
107	ILHAM	ANALIS SENIOR	ORACLE	1
109	RIAN	ANALIS YUNIOR	COBOL	2
109	RIAN	ANALIS YUNIOR	DBASE III+	2
112	FIKA	PEMROGRAM	COBOL	1
112	FIKA	PEMROGRAM	DBASE III+	1
112	FIKA	PEMROGRAM	SYBASE	1

NIP → nama

NIP → jabatan

PK: NIP, nama, jabatan, keahlian

Contoh A Untuk 2NF

NIP	NAMA	JABATAN	KEAHLIAN	LAMA
107	ILHAM	ANALIS SENIOR	COBOL	6
107	ILHAM	ANALIS SENIOR	ORACLE	1
109	RIAN	ANALIS YUNIOR	COBOL	2
109	RIAN	ANALIS YUNIOR	DBASE III+	2
112	FIKA	PEMROGRAM	COBOL	1
112	FIKA	PEMROGRAM	DBASE III+	1
112	FIKA	PEMROGRAM	SYBASE	1

- Skema: (NIP, Nama, Jabatan, Keahlian, Lama)
- Kebergantungan Fungsional:
  - NIP, Nama → Jabatan
  - NIP → Jabatan
  - Nama → Jabatan

Hasil Contoh 2NF

NIP	NAMA	JABATAN
107	ILHAM	ANALIS SENIOR
109	RIAN	ANALIS YUNIOR
112	FIKA	PEMROGRAM

NIP	KEAHLIAN	LAMA
107	COBOL	6
107	ORACLE	1
109	COBOL	2
109	DBASE III+	2
112	COBOL	1
112	DBASE III+	1
112	SYBASE	1

- „NIP → Nama
- „NIP → Jabatan
- „NIP, Keahlian → Lama„

Contoh B Untuk 2NF

No_Pesanan	Tgl_Pesanan	Item	Total
50001	12/05/1997	P1	45.000
50001	12/05/1997	P2	45.000
50001	12/05/1997	P3	45.000
50001	12/05/1997	P4	45.000
50002	12/05/1997	P3	32.500
50002	12/05/1997	P5	32.500
50002	12/05/1997	P6	32.500
50003	13/05/1997	P1	12.000
50003	13/05/1997	P2	12.000

- No_pesanan → Tgl_pesanan
- No_pesanan, Item → Total
- No_pesanan → Total

Hasil Contoh B Untuk 2NF

No_Pesanan	Item
50001	P1
50001	P2
50001	P3
50001	P4
50002	P3
50002	P5
50002	P6
50003	P1
50003	P2

No_Pesanan	Tgl_Pesanan	Total
50001	12/05/1997	45.000
50002	12/05/1997	32.500
50003	13/05/1997	12.000

- {NoPesanan, Item}
- {NoPesanan}
- No_pesanan → tgl_pesanan
- No_pesanan → total

Bentuk Normal Ketiga (3NF)

- Diubah ke 2NF dengan melakukan dekomposisi terhadap relasi tersebut.
- Dapat dilakukan dengan menggambarkan diagram dependensi fungsional terlebih dahulu.
- Berdasarkan diagram ini, relasi dalam bentuk 1NF dapat dipecah ke dalam sejumlah relasi.

Contoh 3NF

Nomor_Pesanan	Nomor_Urut	Kode_Item	Nama_Item
50001	0001	P1	Pensil
50001	0002	P2	Buku Tulis
50001	0003	P3	Penggaris
50001	0004	P4	Penghapus
50002	0001	P3	Penggaris
50002	0002	P5	Pulpen
50002	0003	P6	Spidol
50003	0001	P1	Pensil
50003	0002	P2	Buku Tulis

- „PK:{ nomor_pesanan, nomor_urut}
- „Nomor_pesanan, Nomor_urut → Kode_Item
- „Nomor_pesanan, Nomor_urut → Nama_Item
- „Kode_tem → Nama_Item
- „Nomor_pesanan, nomor_urut → kode_item → nama_item

Hasil Contoh 3NF

Nomor_Pesanan	Nomor_Urut	Kode_Item
50001	0001	P1
50001	0002	P2
50001	0003	P3
50001	0004	P4
50002	0001	P3
50002	0002	P5
50002	0003	P6
50003	0001	P1
50003	0002	P2

Kode_Item	Nama_Item
P1	Pensil
P2	Buku Tulis
P3	Penggaris
P4	Penghapus
P5	Pulpen
P6	Spidol

- No_pesanan, nomor_urut → kode_item
- Kode_item → nama_item

Bentuk Normal Boyce-Codd (BCNF)

- 3NF mencukupi apabila tabel hanya memiliki satu key. Apabila tabel memiliki lebih dari satu key, maka bisa terjadi masalah. Karena itu diperlukan BCNF.
- Suatu relasi disebut memenuhi bentuk normal Boyce-Codd jika dan hanya jika
  - sudah dalam bentuk 3NF, dan
  - semua penentu (determinan) adalah kunci kandidat (atribut yang bersifat unik)
- BCNF merupakan bentuk normal sebagai perbaikan terhadap 3NF; suatu relasi yang memenuhi BCNF selalu memenuhi 3NF tetapi tidak sebaliknya.

Bentuk Normal Boyce-Codd (BCNF)

- „Cara mengkonversi relasi yang telah memenuhi 3NF ke BCNF:
  - carilah semua penentu (determinan)
  - bila terdapat penentu yang bukan berupa kunci kandidat, pisahkan relasi tersebut dan buat penentu tersebut sebagai kunci primer.„

Bentuk Normal Boyce-Codd (BCNF)

- **4 Langkah:**
  1. Tentukan semua kunci kandidat pada tabel tsb.
  2. Tentukan semua dependensi fungsional pada tabel tsb.
  3. Tentukan apakah semua penentu merupakan kunci kandidat?
  4. Dekomposisi table tsb.„

Contoh BCNF: Tabel Enrolment

Dosen	Semester	Kuliah	Sesi	Kehadiran
Joe	1/2005	COBOL	1	35
Jeni	1/2005	MATH	1	40
Garin	2/2005	UNIX	1	33
Jeni	1/2005	MATH	2	42
Garin	2/2005	UNIX	2	47
Joe	1/2005	COBOL	2	50
Joe	1/2005	COBOL	3	12
Joe	2/2005	MATH	1	50

Contoh Kasus BCNF

- Pada contoh tabel tsb, setiap dosen hanya mengajar satu mata kuliah setiap semester namun dapat memiliki beberapa sesi.
- Jadi,
  - Dosen, semester → mata kuliah
  - Mata kuliah, semester → dosen
- Kunci Kandidat:
  - {dosen, semester, sesi}
  - {mata kuliah, semester, sesi}
- Kita masih memiliki masalah redudansi karena dosen untuk setiap mata kuliah masih disimpan lebih dari sekali. Karena itu diperlukan BCNF.

Hasil Contoh BCNF

Semester	Kuliah	Sesi	Kehadiran
1/2005	COBOL	1	35
1/2005	MATH	1	40
2/2005	UNIX	1	33
1/2005	MATH	2	42
2/2005	UNIX	2	47
1/2005	COBOL	2	50
1/2005	COBOL	3	12
2/2005	MATH	1	50

Dosen	Semester	Kuliah
Joe	1/2005	COBOL
Jeni	1/2005	MATH
Garin	2/2005	UNIX
Joe	2/2005	MATH

- „{Semester, Kuliah, Sesi}
- „Semester, kuliah, sesi → kehadiran

- „{Dosen, Semester}
- „Dosen, semester → Kuliah

Bentuk Normal Keempat (4NF)

- Bentuk normal keempat berkaitan dengan sifat Ketergantungan Banyak Nilai (Mutlivalued Dependency) pada suatu tabel yang merupakan pengembangan dari Ketergantungan Fungsional.

Bentuk Normal Kelima (5NF)

- Bentuk tahap kelima (nama lain dari Projeck-Join Normal Form/PJNF) berkenaan dengan Ketergantungan Relasi antar Tabel (Join Dependency).

Langkah-langkah Normalisasi