NORMALISASI BASIS DATA MODEL RELASIONAL DAN IMPLEMENTASINYA DENGAN ORACLE POWER OBJECT

TUGAS AKHIR
Diajukan Sebagai Salah Satu Syarat
Dalam Menempuh Ujian Sarjana Program Strata 1
Teknik Informatika, Institut Teknologi Indonesia

OLEH :
QUANDRIARTO PRASETIYOAEDI
915920033 / 923206716750031

TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI INDONESIA
SERPONG
1998

NORMALISASI BASIS DATA MODEL RELASIONAL DAN IMPLEMENTASINYA DENGAN ORACLE POWER OBJECT

TUGAS AKHIR

Diajukan Sebagai Salah Satu Syarat

Dalam Menempuh Ujian Sarjana Starata – 1

Teknik Informatika, Institut Teknologi Indonesia

OLEH :

QUANDRIARTO PRASETIYOAEDI

015920033 / 9293206716750031

Disahkan Oleh :

Ketua Jurusan Teknik Informatika	Pembimbing
DR. – Ing. Kondar Siahaan	**DR. – Ing. Kondar Siahaan**

ABSTRAKSI

Basis data model relasional adalah model penyajian basis data yang dibangun dari teori – teori relasi matematika. Diperkenalkan pertama kali oleh ilmuwan komputer **E. F. Codd** dan **C. J. Date** pada tahun 1970-an.

Dalam aplikasinya, bentuk relasi ini disajikan dalam bentuk tabel – tabel data. Dimana setiap tabel terdiri atas beberapa baris data yang disebut *record* dan setiap *record* terdiri atas beberapa kolom yang disebut *field*. Tiap satu *record* merupakan satu kesatuan data yang dapat memberikan informasi kepada pengguna sedangkan setiap kesatuan *field* terdiri atas himpunan data yang mempunyai tipe dan sifat data yang sama.

Hal terpenting dari basis data model relasional adalah proses normalisiasi. Proses normalisasi ini digunakan untuk meminimalisasi kasus *anomaly* (keadaan dimana pada suatu relasi data tidak dapat disisipkan kedalam relasi, proses penghapusan data tidak efisien atau kesulitan dalam memperbaharui data), yang mungkin terjadi pada suatu basis data, dengan jalan pemisahan atau mendekomposisikan suatu relasi menjadi sub relasi sehingga menjadi lebih sederhana dan terdiri dari hanya relasi – relasi yang terkecil secara struktural dan mengandung *field* yang lebih sedikit dari aslinya.

KATA PENGANTAR

Salah satu syarat untuk mendapatkan gelar sarjana program Strata – 1 Teknik Informatika, Institut Teknologi Indonesia dengan pembuatan tugas akhir. Tujuan pembuatan tugas akhir ini adalah untuk mengetahui sampai sejauh mana mahasiswa dapat menerapkan ilmu dan mengaplikasikan ilmu yang telah didapat dibangku kuliah serta mengekspresikannya kedalam bentuk tulisan. Tugas akhir tersebut nantinya harus dapat dipertanggungjawabkan dihadapan tim penguji.

Untuk hal tersebutlah maka penulis melakukan pembuatan tugas akhir, dengan judul

NORMALISASI BASIS DATA MODEL RELASIONAL
DAN IMPLEMENTASINYA DENGAN
ORACLE POWER OBJECT

Alasan penulis memilih judul tersebut didasari kenyataan, bahwa basis data model relasional adalah basis data yang paling banyak digunakan dalam basis data modern dan Oracle, yang diproduksi Oracle Corporation, adalah salah satu perangkat lunak yang menggunakan model relasional untuk sistem manajemen basis datanya. Oracle tidak hanya memproduksi sistem manajemen basis data untuk satu jenis komputer saja, melainkan juga untuk beberapa jenis komputer dan sistem operasi yang berbeda.

Konsep Basis data model relasional merupakan konsep matematika relasi (aljabar relasional dan kalkulus relasional). Pada sistem relasional kita dapat mendefinisikan suatu relasi, menyimpan relasi, mengambil relasi, menampilkan relasi, mengisi relasi, mengisi tupel – tupel relasi serta mengoperasikan relasi berdasarkan operasi – operasi matematika.

Bahasan terpenting dalam tugas akhir ini adalah normalisasi (*normalization*) pada basis data model relasional, sistem relasi dan operasi – operasinya. Bagian ini merupakan batasan masalah pada tugas akhir ini.

Implementasi tugas akhir ini menggunakan SQL (*structured query language*) Oracle, dimana perangkat lunak Oracle yang digunakan hanya berfungsi sebagai sarana untuk mengimplementasikan tabel – tabel hasil normalisasi menjadi bentuk nyata sebuah program basis data. Perangkat lunak Oracle yang digunakan pada tugas akhir adalah Oracle Power Object v1.0, dengan struktur tabel database Blaze.

Sistematika penulisan tugas akhir ini adalah :

- Bab 1 yang merupakan bab pendahuluan menjabarkan secara garis besar konsep basis data, komponen basis data, penyajian basis data, kemandirian data serta model – model basis data.
- Basis data model relasional akan dibahas pada Bab 2, yang meliputi aljabar relasional, kalkulus relasionai dan *structured query language*.
- Ketergantungan (*dependency*) dan kunci (*key*) berada pada Bab 3.
- Inti dari tugas akhir ini berada pada Bab 4 Normalisasi (*normalization*) Basis Data Model Relasional, yang akan menjelaskan bentuk – bentuk normalisasi disertai dengan contoh aplikasinya.
- Bab 5 akan membahas implementasi normalisasi basis data model relasional dari hasil relasi bentuk – bentuk normalisasi pada Bab 4, dengan menggunakan perangkat lunak Oracle Power Object v1.0.
- Analisa dan kesimpulan normalisasi untuk sistem basis data model relasional berada pada bab penutup yaitu Bab 6.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR ISTILAH

ENTITY

Entity adalah kejadian atau konsep yang informasinya direkam.

ATRIBUT

Setiap kolom dari suatu relasi disebut sebagai atribut suatu relasi.

NILAI DATA (DATA VALUE)

Nilai data adalah data aktual yang disimpan pada tiap data elemen atau atribut.

RECORD / TUPEL

Kumpulan elemen – elemen yang saling berkaitan memberikan informasi tentang *entity* secara lengkap, dimana satu *record* mewakili satu data.

FIELD

Kumpulan *record* sejenis yang mempunyai sifat dan tipe data yang sama.

UCAPAN TERIMA KASIH

Puji dan Syukur atas kehidirat Allah Subhanu Wata'ala, karena berkat taufik dan hidayah-Nya maka penulis dapat menyelesaikan tugas akhir.

Pertama – tama penulis ucapan terimakasih dan penghargaan setinggi – tingginya kepada :

- Ayahanda, Ibunda Santi, Ernuko, Esti dan Tasya, atas doa dan doronganya
- Bapak DR. – Ing. Kondar Siahaan selaku pembimbing Tugas Ahir, pembimbing akademis Teknik Informatika Angkatan 92 Ketua Jurusan Teknik Informatika, Institut Teknologi Indonesia.
- Bapak Ir. Piping Supriyatna, MSc selaku penguji pertama.
- Bapak Ir. Oon Amroni, MSc selaku penguji kedua.
- Bapak Dipl. Math. Syafwan Syarif selaku penguji ketiga.
- Semua teman – teman sesama citivitas akademika Institut Teknologi Indonesia atas bantuan yang diberikan kepada penulis

BAB 1
KONSEP DASAR BASIS DATA

1.1. BASIS DATA

Sistem basis data adalah suatu sistem yang menyusun dan mengelola data – data menggunakan komputer, dengan tujuan untuk menyimpan / merekam serta memelihara data sebagai operasional lengkap sebuah organisasi / perusahaan, sehingga mampu menyediakan informasi yang diperlukan pemakai untuk kepentingan proses pengambilan keputusan.

Karena data adalah suatu sumber daya, maka data perlu dikelola. Proses ini disebut manajemen data. Kegiatan manajemen data mencakup :

- **Pengumpulan data**, data yang diperlukan dikumpulkan dan dicatat dalam suatu formulir yang disebut sumber dokumen (*source document*) yang berfungsi sebagai masukkan bagi sistem.
- **Integritas dan pengujian**, data tersebut diperiksa untuk meyakinkan konsistensi dan akurasinya berdasarkan suatu aturan – aturan dan masalah – masalah yang telah ditentukan sebelumnya.
- **Penyimpanan**, data disimpan pada suatu media seperti *magnetic tape* atau *magnetic disk*.
- **Pemeliharaan**, ketika suatu data baru ditambahkan, maka data yang tidak diperlukan lagi dirubah atau dihapus, agar selalu diperoleh sumber daya data yang terbaru.
- **Keamanan**, data dijaga untuk mencegah penghancuran, kerusakan atau penyalahgunaan.
- **Organisasi**, data disusun sedemikian rupa untuk memenuhi kebutuhan informasi pemakai.
- **Pengambilan**, data selalu tersedia bagi pemakai untuk digunakan.

1.2. KOMPONEN POKOK BASIS DATA

Sistem basis data mempunyai empat komponen pokok, yaitu data, perangkat keras, perangkat lunak dan pemakai.

1.2.1. Data

Data didalam sebuah sistem basis data :

- Disimpan secara terintegrasi, artinya basis data merupakan kumpulan dari berbagai macam file dari aplikasi – aplikasi yang berbeda, yang disusun dengan cara menghilangkan bagian – bagian yang rangkap.
- Data dapat dipakai secara bersama – sama, artinya masing – masing bagian dari basis data dapat diakses oleh pemakai dalam waktu yang bersamaan untuk aplikasi yang berbeda

1.2.2. Perangkat Keras

Perangkat keras terdiri dari semua peralatan perangkat keras yang digunakan untuk mengelola sistem basis data, berupa :

- Peralatan yang menyimpanan basis data, peralatan masukkan dan keluaran, peralatan kontrol, saluran masukkan dan keluaran dan lain sebagainya.
- Prosesor dan memori utama yang mendukung jalannya perangkat lunak basis data.
- Peralatan – peralatan komunikasi data.

1.2.3. Perangkat Lunak

Perangkat lunak berfungsi sebagai perantara antara pemakai dengan data fisik. Perangkat lunak sistem basis data dapat berupa :

- Data base management sistem (DBMS) adalah sistem yang menangani akses pada basis data sehingga pemakai tidak perlu memikirkan proses penyimpanan dan pengelolaan data.
- Program – program aplikasi dan prosedur – prosedur.

1.2.4. Pemakai

Pemakai basis data dibagi atas tiga kelompok, yaitu :

- Programmer, yaitu orang / tim yang bertugas membuat program aplikasi yang mengakses basis data dengan menggunakan bahasa pemrograman.
- End-user, yaitu orang yang mengakses basis data melalui terminal dengan menggunakan bahasa terstruktur (*query-language*) atau program aplikasi yang dibuat programmer.
- Data base administrator (DBA), yaitu orang / tim yang bertugas mengelola sistem basis data secara keseluruhan.

1.3. PENYAJIAN DATA PADA BASIS DATA

Ada tiga jenis penyajian data didalam sistem basis data, yaitu :

1. **Data operasional,** merupakan data dari suatu organisasi yang disimpan didalam basis data.
2. **Data masukkan,** merupakan data dari luar sistem yang dimasukkan melalui peralatan input, yang dapat merubah data operasional.
3. **Data keluaran,** berupa laporan melalui peralatan output, yang merupakan hasil proses dari dalam sistem yang mengakses data operasional.

1.4. KEMANDIRIAN DATA

Era penggunaan komputer yang ada sebelum konsep basis data ditandai dengan pengulangan data *(redundancy data)*, ketergantungan data *(data dependency)* dan kepemilikan data yang tersebar *(diffused data ownership)*.

Konsep basis data adalah suatu koleksi data komputer. yang terintegrasi, diorganisasikan dan disimpan dalam suatu cara yang memudahkan pengambilan kembali. Dua tujuan utama dari konsep basis data adalah meminimumkan pengulangan dan mencapai kemandirian data.

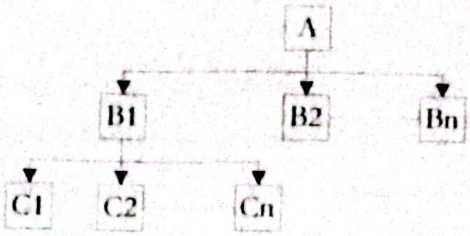
Kemandirian data adalah kemampunan untuk membuat perubahan dalam struktur data tanpa membuat perubahan pada program yang memproses data. Kemandirian data dicapai dengan menempatkan spesifikasi dalam tabel dan kamus yang terpisah secara fisik dari program. Program mengacu pada tabel untuk mengakses data. Perubahan pada struktur data hanya sekali yaitu dalam tabel.

1.5. MODEL – MODEL BASIS DATA

1.5.1. Model Hirarkis

Didalam model hirarkis, suatu hubungan diantara data disajikan menurut struktur data pohon *(tree)*. Data yang direpresentasikan dalam model hirarkis ini harus mempunyai kondisi :

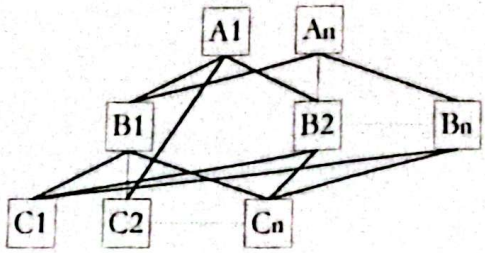
1. Harus terdapat simpul awal yang disebut akar, simpul ini haruslah unik
2. Semua simpul selain simpul akar harus terkait dengan satu dan hanya satu simpul dengan yang tingkatnya lebih tinggi.



Gambar 1.1. Struktur Diagram Model Hirarkis

1.5.2. Model Jaringan

Model Jaringan (*network model*) merupakan perluasan dari model hirarkis. Perbedaan yang mendasar adalah satu simpul dapat dikaitkan dengan lebih dari satu simpul data dan tidak terdapat perbedaan tingkat antara simpul – simpul data tersebut.



Gambar 1.2. Struktur Diagram Model Jaringan

1.5.3. Model Relasional

Didalam Model Relasional, data serta hubungan antar data direpersentasikan oleh sejumlah tabel. Tiap tabel terdiri atas beberapa baris data yang disebut *record* dan beberapa kolom yang disebut *fileld*. Tiap satu *record* merupakan satu kesatuan data yang dapat memberikan informasi kepada pengguna, sedangkan tiap satu kesatuan *field* terdiri atas himpunan data yang mempunyai tipe dan sifat data yang sama.

Atribut Ke - 1	Atribut Ke - 2	.....	Atribut Ke – n
a_{11}	a_{12}		a_{1n}
a_{21}	a_{22}		a_{2n}
a_{m1}	a_{m2}		a_{mn}

Tabel 1.1. Struktur Tabel Model Relasional

BAB 2
BASIS DATA MODEL RELASIONAL

Pada awal tahun 1970-an **E.F. Codd** dan **C.J. Date** dari IBM secara terpisah mengembangkan suatu pendekatan untuk menetapkan hubungan antar catatan yang tidak harus dinyatakan secara eksplisit. Pendekatan Codd dan Date dinamakan struktur relasional dengan menggunakan hubungan yang implisit, yaitu hubungan yang dapat dinyatakan secara tidak langsung dari catatan data yang telah ada.

Basis data model relasional banyak digunakan pada sistem basis data modern sebagai dasar model utama dari sistem basis data komersial. Model ini mempunyai sejumlah konsep sederhana yang ditujukan untuk menyimpan data didalam sebuah basis data, bersama dengan sejumlah operasi yang digunakan untuk mengoperasikan atau memanipulasi data – data tersebut.

Beberapa keuntungan dengan penggunaan basis data model relasional adalah :

- Data yang disajikan mudah dibaca dan dipahami, karena data disajikan dalam bentuk tabel sehingga proses pengambilan keputusan menjadi lebih cepat.
- Informasi data yang terdapat dalam tabel – tabel yang berlainan dapat diperoleh sekaligus melalui proses manipulasi data yang terdapat pada model realsional. Hal ini disebabkan diantara tabel – tabel saling berelasi.
- Aksesibilitas data lebih cepat karena secara fisikal setiap datanya disimpan dalam ruang memori berstruktur data sederhana atau ruang memori yang struktur datanya tidak saling terkait satu dengan yang lain.
- Hampir semua perangkat lunak basis data yang terdapat dipasaran mendukung model relasional, sehingga semakin memudahkan manajemen data dan pendistribusian data.

Bahasa *query* adalah bahasa yang digunakan oleh pengguna untuk mendapatkan informasi dari basis data. Bahasa *query* umumnya berupa bahasa tingkat tinggi yang levelnya lebih tinggi dari bahasa pemrograman standar. Bahasa *query* terbagi atas dua kategori, yaitu :

- **Bahasa Query Prosedural.** Pada bahasa *query* prosedural, interaksi antara pengguna dengan basis data harus dilakukan dengan memasukkan serangkaian instruksi agar sistem dapat melaksanakan serangkaian operasi yang berurutan dalam rangka memperoleh hasil yang dikehendaki.
- **Bahasa Non-Prosedural.** Pada bahasa *query* non-prosedural, pengguna memperoleh hasil yang dinginkan tanpa memberikan instruksi – instruksi tertentu kedalam sistem.

2.1. ALJABAR RELASIONAL

Aljabar relasional adalah sebuah *procedural query language*. Untuk mengoperasikan basis data dengan menggunakan aljabar relasional, digunakan operator – operator khusus yang diambil dari operator – operator pada set teori. Operator pada aljabar relasional terbagi atas dua kelompok yaitu operator dasar (*fundamental operation*) dan operator tambahan (*additional operation*).

2.1.2. Operator Dasar

Yang termasuk operator dasar (*fundamental operation*) pada aljabar relasional adalah operator seleksi (*selection*), proyeksi (*project*), hasil kali kartesis (*cartesian product*), gabungan (*union*) dan selisih (*set difference*).

1. **Seleksi (σ)**, operasi seleksi akan menyeleksi tupel – tupel pada sebuah relasi, yaitu tupel – tupel yang memenuhi syarat – syarat yang telah ditentukan sebelumnya. Operasi seleksi dituliskan dengan bentuk umum $\sigma_t (R)$

2. **Proyeksi (Π)**, operator proyeksi beroperasi pada sebuah relasi yang akan membentuk relasi yang baru dengan mengkopi atribut – atribut dan domain – domain dari relasi tersebut berdasarkan argumen – argumen pada operasi tersebut. Operasi proyeksi ditulis dengan bentuk umum $\Pi_{a1,a2,\ldots,an}$ **(R)**
3. **Hasil Kali Kartesis (X)**, operator ini merupakan operator untuk operasi *binary*, yaitu operator yang berelasi pada dua relasi. Operasinya ditulis dengan bentuk umum **R1 X R2.**
4. **Gabungan (∪)**, operator gabungan juga digunakan untuk operasi *binary*. Operasi ini akan membentuk relasi baru dengan tupel – tupel yang terdapat di relasi R1 atau R2 atau tupel – tupel yang terdapat di kedua relasi tersebut. Operasi gabungan ditulis dengan bentuk umum **R = R1 ∪ R2**
5. **Selisih (−)**, operator ini juga merupakan operasi *binary* yang akan membentuk relasi baru dengan tupel – tupel yang terdapat di R1, tidak terdapat direlasi R2. Operasi selisih ditulis dengan bentuk umum **R = R1 - R2.**

2.1.2. Operator Tambahan

Yang termasuk di dalam operator tambahan (*additional operation*) pada aljabar relasional adalah operator irisan (*set intersection*), *theta join*, *natural join* dan pembagian (*division*). :

1. **Irisan**, merupakan operarator *binary* untuk membentuk sebuah relasi baru dengan tupel – tupel yang berasal dari kedua relasi yang dihubungkan. Misal R1 adalah relasi dengan elemen E1, E2, ……, En dan R2 adalah relasi dengan elemen T1, T2, ……, Tn. Maka operasi irisan **R = R1 ∩ R2**, akan menghasilkan relasi dengan R elemen yang terdapat di R1 juga terdapat di R2. Operator irisan tidak termasuk operator dasar karena operator tersebut dapat dituliskan dengan menggunakan operator selisih **R1 ∩ R2 = R1 – (R1 – R2).**
2. **Theta Join (0)**, merupakan operator yang menggabungkan operator hasil kali kartesis dengan operator seleksi. Bila R1 dan R2 adalah relasi,

sedangkan θ adalah predikat, maka operator *theta join* mempunyai bentuk umum

$R1 \theta R2 = \sigma_{\theta}(R1 \times R2)$

3. **Natural Join,** merupakan operator yang melakukan operasi penggabungan terhadap tupel – tupel dari relasi – relasi yang dioperasikan. Misal R1 dan R2 adalah relasi yang mempunyai beberapa kolom yang sama, maka hasil operasi *natural join* adalah operasi hasil kali kartesis dari beberapa tupel – tupel di relasi R1 dan R2 yang nilai atribut sama.
4. **Pembagian**, merupakan operasi pembagian atas tupel – tupel dari dua relasi. Misal R1 adalah sebuah relasi dengan aritas R dan R2 adalah sebuah relasi dengan aritas S, maka operator pembagian dengan simbol ÷ akan menghasilkan relasi baru dengan aritas R – S.

2.1. KALKULUS RELASIONAL

Kalkulus relasional merupakan bahasa *query* non-prosedural. Untuk mendapatkan informasi yang diinginkan dalam sebuah basis data, pengguna hanya memberikan deskripsi formalnya saja, tanpa spesifikasi tentang bagaimana urutan – urutan proses untuk mendapatkan informasi tersebut.

Terdapat dua bentuk kalkulus relasional yaitu kalkulus tupel relasional (*relational tuple calculus*) dan kalkulus domain relasional (*relational domain calculus*).

2.2.1. Kalkulus Tupel Relasional

Pada kalkulus tupel relasional setiap variabelnya merepresentasikan nilai – nilai dari tupel-nya. *Query* pada sebuah tupel direpresentasikan sebagai $\{t \mid P(t)\}$, yaitu sejumlah tupel dari seluruh tupel t, dimana predikat P adalah benar untuk tupel t tersebut.

Simbol – simbol lain yang terdapat pada kalkulus tupel relasional adalah :

- t[A] : menyatakan value tupel t pada atribut A
- t ∈ r : menyatakan tupel t berada pada relasi r.

1. **There exists**, bila diinginkan atribut tertentu saja dari sebuah tupel t, maka digunakan *there exists* dengan menggunakan simbol ∃, dimana notasi logika matematikanya mempunyai bentuk umum **∃ t (Q (t)),** yaitu terdapat sebuah tupel *t*, dimana predikat Q (t) adalah *true*.
2. **Union**, pada operasi gabungan dibutuhkan dua buah *there exists clause* yang dihubungkan oleh operator OR (∨)
3. **Intersection**, pada operasi irisan digunakan operator *and* (∧) serta *there exists clausa*.
4. **Set Difference**, pada operasi pengurangan digunakan operator *not* (¬) serta *there exists clause*.
5. **Call**, digunakan untuk operasi pembagian, dimana ekspresinya pada ekspresi matematika ditulis dengan ∀ t (Q (t)). Dimana predikat Q adalah *true* untuk semua tupel t.

Pada bentuk umum $\{t \mid P(t)\}$, t adalah sebuah variabel bebas. Sedangkan untuk notasi S ∈ R, maka S adalah tupel variabel atau *bound-variable*. Kalkulus tupel relasional terbentuk dari atom – atom, dimana sebuah atom dapat berupa :

- S ∈ R, dimana S adalah tupe variabel dan r adalah relasi.
- S[x] θ U[y], dimana S dan U adalah tupel variabel dan x adalah atribut pada tupel S dan Y atribut pada tupel U.
- θ adalah operator – operator <, ≤, =, ≠, >, ≥..
- S[x] θ c, diman c adalah konstanta.

2.2.2. Kalkulus Domain Relasional

Operator kedua dari kalkulus relasional adalah kalkulus domain relasional, yaitu bentuk yang menggunakan *domain* (judul kolom) sebagai variabel. Operator ini pada dasarnya sangat erat hubungannya dengan kalkulus tupel relasional.

Kalkulus domain relasional diekspresikan dalam bentuk $\{<x_1,x_2,\ldots,x_n> | P(x_1,x_2,\ldots,x_n)\}$

Seperti halnya kalkulus tupel relasional, sebuah atom dalam kalkulus domain relasional harus mempunyai satu dari bentuk berikut :

- $<x_1,x_2,\ldots,x_n> \in r$, dimana r adalah tabel yang mempunyai n kolom dan $x_1,x_2,\ldots,x_n$ variabel domain atau konstanta domain.
- S[x] θ U[y], dimana S dan U adalah tupel variabel dan x adalah atribut pada tupel S dan Y atribut pada tupel U.
- θ adalah operator – operator $<, \leq, =, \neq, >, \geq$.
- S[x] θ c, diman c adalah konstanta.

2.3. STRUCTURED QUERY LANGUAGE

Structured query language yang biasa disingkat dengan SQL dikembangkan oleh lembaga riset IBM di laboratorium San Jose. SQL merupakan bahasa standart untuk mengakses basis data yang bersifat bahasa *query* non-prosedural karena bahasa ini tidak memiliki perintah pengulangan (*looping*) maupun percabangan (*condition*).

Bahasa ini banyak digunakan oleh basis data model relasional karena kemampuannya untuk mendefinisikan *database object* (*table, view, index, sequence*), *constraint* (*primary key, foreign key, check, unique, not null*) dan kemampuannya untuk memanipulasi *database object*.

SQL terdiri dari tiga perintah dasar, yaitu :

1. **Select**, merupakan operasi *projection* dari aljabar relasional, digunakan untuk mendapatkan atribut – atribut yang diinginkan dan ditampilkan pada relasi sebagai hasil *query*.
2. **Form**, merupakan relasi – relasi yang akan dioperasikan.
3. **Where**, merupakan predikat /persyaratan yang berkaitan dengan operasi *selection* pada aljabar relasional.

Bentuk umum instruksi SQL adalah :

Select $A_1, A_2, A_3, \ldots\ldots\ldots, A_n$
Form $R_1, R_2, R_3, \ldots\ldots\ldots, R_n$
Where P

Pada SQL untuk menghilangkan duplikasi pada tupel – tupel digunakan instruksi *distinct*. Sedangkan untuk operasi gabungan, irisan dan pengurangan digunakan perintah *union*, *intersect* dan *minus*. SQL tidak mempunyai operator yang secara langsung merepresentasikan *natural join*, tetapi dilaksanakan oleh gabungan operasi *cartesian product*, *selection* dan *projection*.

BAB 3
KETERGANTUNGAN DAN KUNCI

3.1. KETERGANTUNGAN

3.1.1. Ketergantungan Fungsional

Ketergantungan fungsional adalah keterkaitan antara dua buah atribut, dituliskan dengan cara $A \rightarrow B$. Dimana A dan B adalah atribut – atribut pada suatu relasi r.

3.1.2. Ketergantungan Parsial dan Penuh

Jika $X \rightarrow A$ adalah ketergantungan fungsional *non trivial* ($A \not\subset X$), maka atribut A adalah tergantung secara parsial pada X jika $Y \rightarrow A$ ada untuk $Y \rightarrow A$ ada untuk $Y \subset X$. Dalam hal ini $X \rightarrow A$ adalah ketergantung parsial.

Jika $X \rightarrow A$ adalah ketergantungan fungsional tak trivial dan $\exists\ Y \subset X$, sehingga $Y - / \rightarrow A$, maka adalah $X \rightarrow A$ ketergantungan penuh.

3.1.3. Ketergantungan Transitif

Jika X dan Y atribut tidak kosong, subset dari U, $\exists\ A \subset U$, maka atribut dari A adalah bergantung secara transitif pada X melalui Y jika memenuhi semua kondisi $(X \rightarrow Y)$, $(Y - / \rightarrow X)$, $(Y \rightarrow A)$ dan $(A \not\subset XY)$.

Jika suatu ketergantungan fungsional $X \rightarrow A$ memenuhi kondisi diatas, maka $X \rightarrow A$ adalah ketergantungan transitif.

3.1.4. Ketergantungan Banyak Nilai

Pada ketergantungan fungsional, setiap nilai dari determinan A hanya menentukan satu nilai ketergantungan B, digambarkan sebagai A → B. Tetapi ada kemungkinan terjadi nilai determinan A tidak secara unik menentukan nilai ketergantungan B.

3.1.5. Ketergantungan Join

3.1.5.1. Dekomposisi

Jika R adalah relasi dan R_1, R_2,....,R_n adalah himpunan proyeksi R, sehingga R_1, R_2,....,R_n dapat digabungkan melalui atribut – atributnya untuk mereproduksi R.

Jika $t \in R$ adalah tupel sembarang, t mengembangkan t_i pada setiap R_i, $i = 1,\ldots,n$, maka sejumlah n tupel t_i dapat digabung kembali untuk membangun kembali t dengan menggunakan join, R_1 join R_2 ..... join R_n Dengan kata lain kita akan selalu mendapatkan $R \subset R_1$ join R_2 .... join R_n Sehingga, setiap tupel di R akan tampil pada join sebelah kanan. Relasi $R_1,\ldots,R_n$ disebut dekomposisi dari relasi R.

3.1.5.2. Lossy Join dan Lossless Join

Jika suatu relasi didekomposisi menjadi dua atau lebih relasi – relasi yang lebih kecil, maka dengan memperlakukan operasi *natural join (x)* kepada relasi – relasi tersebut, hasilnya akan berupa relasi baru yang mungkin mempunyai ukuran tupel yang lebih banyak dari ukuran aslinya.

Penggunaan relasi baru tersebut dalam sistem basis data akan menghasilkan informasi yang salah (*lossy information*) khususnya dalam pengambilan keputusan.

Suatu relasi mengalami hilangnya informasi akibat operasi join (*lossy join*), jika untuk suatu relasi R, dan R_1, R_2,…,R_n adalah hasil dekomposisi R memenuhi hubungan $R \subseteq R1xR2x\ldots xRn$.

3.1.5.3. Ketergantungan Join

Jika R suatu relasi dan A,B,C…..,Z adalah sembarang himpunan bagian dari himpunan atribut – atribut R, maka R dikatan R memenuhi ketergantungan join, dinotasikan sebagai *(A,B,…,Z), jika dan hanya jika R sama dengan hasil dari proyeksi join A,B,C…,Z.

3.2. KUNCI

Kunci adalah sebuah atribut (atau sejumlah atribut) yang mengidentifikasikan *record* / baris dalam sebuah relasi dengan cara yang unik. Setiap relasi harus mempunyai kunci dan nilai dari kunci tersebut harus unik, yang berarti setiap *record* / baris pada sebuah relasi harus berbeda.

Ada beberapa tipe kunci yaitu kunci kandidat, kunci utama, kunci alternatif, kunci komposit dan kunci tamu (*foreign key*).

3.1.1. Kunci Kandidat

Kunci kandidat adalah atribut – atribut yang menjadi determinan dan dapat dijadikan identitas *record*. Pada sebuah relasi biasanya bisa terdapat satu atau lebih kunci kandidat.

3.1.2. Kunci Utama

Kunci utama adalah kunci kandidat yang menjadi identitas *record* karena dapat mengidentifikasi *record* secara unik dan juga dapat mewakili setiap kejadian dari suatu *entity*.

3.1.3. Kunci Alternatif

Kunci alternatif adalah kunci kandidat yang tidak dijadikan kunci utama.

3.1.4. Kunci Komposit

Kunci komposit adalah kunci yang terdiri dari dua atribut atau lebih, dimana atribut – atribut tersebut bila berdiri sendiri tidak menjadi identitas *record*, tetapi bila dirangkaikan menjadi satu kesatuan akan dapat mengidentifikasikan *record* secara unik.

3.1.5. Kunci Tamu

Kunci tamu adalah bukan kunci atribut pada sebuah relasi yang juga menjadi kunci utama atribut pada relasi lainnya. Kunci tamu digunakan sebagai penghubung antar *record* dari kedua relasi tersebut.

BAB 4

NORMALISASI BASIS DATA MODEL RELASIONAL

Normalisasi merupakan sebuah teknik dalam logikal desain sebuah basis data, yang digunakan sebagai proses pengelompokan atribut dari suatu relasi sehingga membentuk *well-structured relation*.

Well-structured relation adalah sebuah relasi yang jumlah kerangkapan datanya sedikit serta memberikan kemungkinan bagi pengguna untuk melakukan penyisipan *(insert)*, pengahapusan *(delete)* dan modifikasi *(modify)* terhadap baris – baris data pada suatu relasi, tanpa berakibat terjadinya kesalahan (*error*) atau inkonsistensi data (*anomaly*) yang disebabkan oleh operasi – operasi tersebut.

Nomor Mahasiswa	Kode Matakuliah	Biaya Kuliah
92150001	IF – 206	75000
92150002	IF – 306	100000
92150003	IF – 206	75000
92150004	IF – 403	150000
92150005	IF – 306	100000
92150006	IF – 406	50000

Tabel 4.1. Relasi Biaya – Matakuliah

Pada relasi diatas, setiap biaya kuliah selalu berulang pada setiap mahasiswa. Akibatnya akan menyebabkan inkonsistensi data bila dilakukan perubahan (*update*) pada relasi tersebut, yang disebut dengan anomali. Terdapat tiga macam anomali, yaitu :

- Anomali penyisipan (*insertion anomaly*), yaitu inkonsistensi data yang terjadi sebagai akibat operasi penyisipan pada sebuah relasi. Bila ada matakuliah baru yang akan diajarkan (IF – 415), maka matakuliah tersebut tidak dapat disisipkan kedalam relasi sampai ada mahasiswa yang mengambil matakuliah tersebut.
- Anomali penghapusan (*deletion anomaly*), yaitu inkonsistensi data yang terjadi sebagai akibat operasi penghapusan pada sebuah relasi. Bila mahasiwa dengan nomor mahasiswa 92150006, membatalkan matakuliah IF – 406, akan mengakibatkan hilangnya informasi mengenai matakuliah tersebut.
- Anomali perubahan (*update anomaly*), yaitu inkonsistensi data yang terjadi sebagai akibat operasi perubahan pada sebuah relasi. Bila biaya matakuliah dinaikkan, maka harus dilakukan perubahan pada *record – record* mahasiswa, agar data tetap konsisten.

Oleh karena itu harus dilakukan normalisasi terhadap relasi tersebut. Berdasarkan teori normalisasi, maka relasi tersebut dipecah menjadi dua relasi sebagai berikut :

Nomor Mahasiswa	Kode Matakuliah
92150001	IF – 206
92150002	IF – 306
92150003	IF – 206
92150004	IF – 403
92150005	IF – 306
92150006	IF – 406

Tabel 4.2. Relasi Mahasiswa – Matakuliah

Kode Matakuliah	Biaya Kuliah
IF – 206	75000
IF – 306	100000
IF – 403	150000
IF – 406	50000

Tabel 4.3. Relasi Matakuliah - Biaya

4.1. NORMALISASI BENTUK PERTAMA (1NF)

Pada tahap normalisasi bentuk pertama, relasi yang belum dinormalisasikan *(unnormalized relation)* dirubah menjadi normalisasi bentuk pertama dengan cara menghilangkan pengulangan yang terdapat pada relasi yang belum dinormalisasikan.

Terdapat sebuah relasi yang belum dinormalisasikan seperti pada tabel 4.4. Agar relasi pada tabel 4.4. menjadi normalisasi bentuk pertama (1NF), hilangkanlah elemen yang sama dan data yang berulang – ulang, sehingga normalisasi bentuk pertama (1NF) seperti pada tabel 4.5.

Relasi belum bisa dikategorikan sebagai *well-structured relation*, bila didalamnya masih terdapat anomali, sehingga harus dilakukan normalisasi tahap kedua.

Nomor Mahasiswa	Nama Mahasiswa	Alamat	Jurusan	Kode Matakuliah	Sks	Matakuliah	Pengajar	Kelas	Nilai
92320671	Alexander	Jl. Hang Tuah 9	Manajemen Informatika	MI - 306	3	Sistem Informasi Manajemen	Paul	B-104	A
				MI - 406	3	Basis Data	Anton	B-117	B
92320672	Quaxis	Jl. Anggur 10	Teknik Informatika	TI - 306	3	Basis Data	Anton	B-117	A
				TI - 201	4	Mikroprosesor	Denny	H - 301	B
				TI - 206	4	Struktur Data	Joko	C - 309	B

Tabel 4.4. Relasi Yang Belum Dinormalisasikan

Nomor Mahasiswa	Nama Mahasiswa	Alamat	Jurusan	Kode Matakuliah	Sks	Matakuliah	Pengajar	Kelas	Nilai
92320671	Alexander	Jl. Hang Tuah 9	Manajemen Informatika	MI - 306	3	Sistem Informasi Manajemen	Paul	B-104	A
92320671	Alexander	Jl. Hang Tuah 9	Manajemen Informatika	MI - 406	3	Basis Data	Anton	B-117	B
92320672	Quaxis	Jl. Anggur 10	Teknik Informatika	TI - 406	3	Basis Data	Anton	B-117	A
92320672	Quaxis	Jl. Anggur 10	Teknik Informatika	TI - 201	4	Mikroprosesor	Denny	H - 301	B
92320672	Quaxis	Jl. Anggur 10	Teknik Informatika	TI - 206	4	Struktur Data	Joko	C - 309	B

Tabel 4.5. Normalisasi Bentuk Pertama (1 NF)

Bila diteliti secara seksama maka diketahui masih terdapat anomali pada tabel 4.5, yaitu :

- Matakuliah baru tidak dapat dimasukkan, bila tidak mahasiswa yang mengambil matakuliah tersebut → anomali penyisipan.
- Apabila mahasiswa Alexander pindah jurusan, maka harus dilakukan perubahan pada beberapa *record* → anomali perubahan.

Sehingga normalisasi tahap kesatu relasi tersebut belum *well structured* dan harus dilakukan normalisasi tahap kedua untuk membentuk normalisasi tahap yang kedua.

4.2. NORMALISASI BENTUK KEDUA (2NF)

Untuk melakukan normalisasi tahap kedua, sebelumnya dilakukan analisa mengenai ketergantungan fungsional atas atribut – atributnya serta menentukan atribut – atribut yang menjadi kunci utama *(primary key)*.

Nomor Mahasiswa → Nama Mahasiswa, Alamat, Jurusan

Kode Matakuliah → Matakuliah, Sks, Pengajar, Kelas

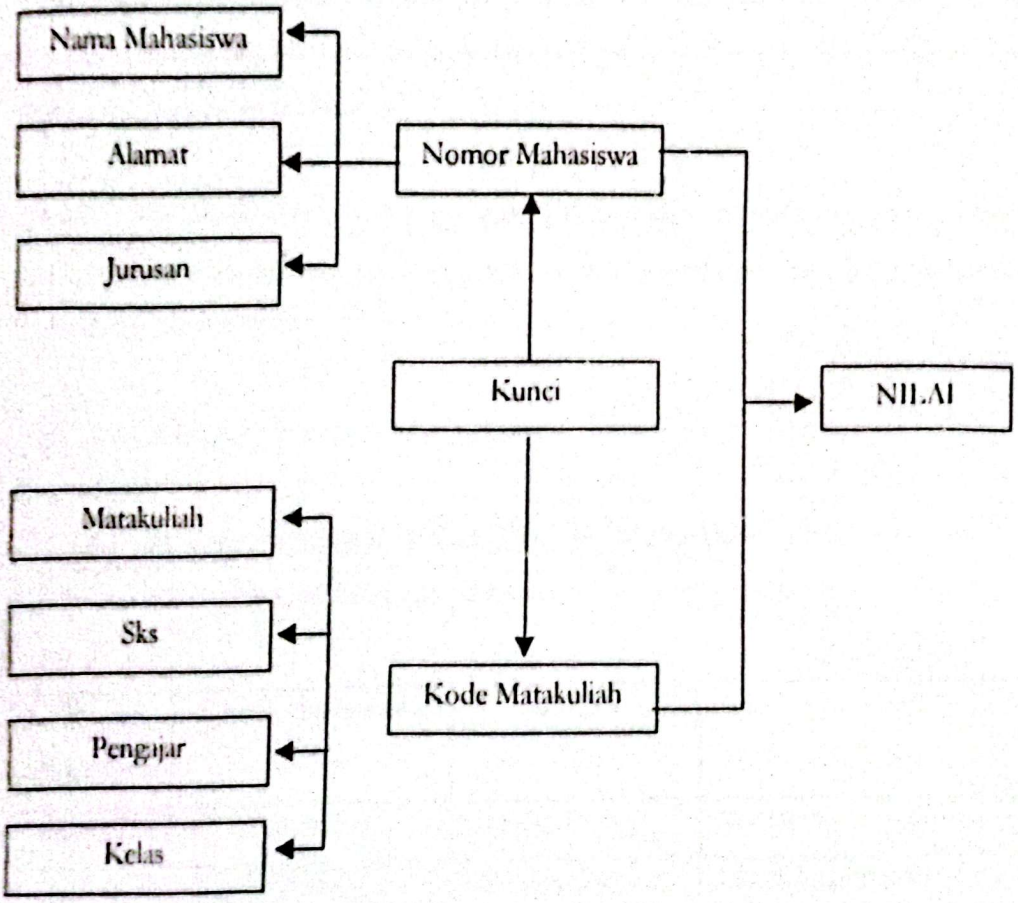
Nomor Mahasiswa, Kode Matakuliah → Nilai

Pengajar → Kelas

Dengan memperhatikan ketergantungan fungsional diatas, dapat disimpulkan bahwa atribut **Nomor Mahasiswa** dan **Kode Matakuliah** menjadi kunci komposit dan merupakan kunci utama pada relasi tersebut.

Kriteria bahwa sebuah relasi sudah merupakan normalisasi bentuk kedua adalah apabila pada relasi tersebut sudah tidak terdapat ketergantungan parsial.

Ketergantungan fungsional yang terjadi dapat digambarkan dalam diagram sebagai berikut :



Gambar 4.1. Diagram Ketergantungan Fungsional Relasi

Dengan memperhatikan data diatas, terlihat bahwa :

- Hanya atribut **Nilai** yang tergantung penuh terhadap kunci komposit **Nomor Mahasiswa** dan **Kode Matakuliah**.

- Sedangkan atribut **Nama Mahasiswa, Alamat** dan **Jurusan** tergantung parsial terhadap kunci **[Nomor Mahasiswa]**.
- Begitupun atribut **Matakuliah, Pengajar** dan **Kelas** tergantung parsial terhadapkunci **[Kode Matakuliah]**.

Berdasarkan kriteria diatas, sebuah relasi sudah bisa dikategorikan menjadi bentuk normal kedua bila sudah tidak terdapat lagi ketergantungan partial dan ketergantungan fungsional pada relasi tersebut.

Untuk menghilangkan ketergantungan partial dan fungsional pada normalisasi bentuk pertama maka relasi tersebut dipecah menjadi beberapa relasi yang baru, yang merupakan normalisasi bentuk kedua.

Pada contoh diagram ketergantungan fungsional diatas didapat :

1. Relasi **Mahasiswa** dengan atribut **[Nomor Mahasiswa]** sebagai kunci atribut **Nama Mahasiswa, Alamat** dan **Jurusan**.

Nomor Mahasiswa	Nama Mahasiswa	Alamat	Jurusan
92150001	Alexander	Jl. Hang Tuah 9	Manajemen Informatika
92150002	Quaxis	Jl. Anggur 10	Teknik Informatika

Tabel 4.6. Relasi Mahasiswa

2. Relasi **Pengajar** dengan atribut **[Kode Matakuliah]** sebagai kunci atribut **Matakuliah, Pengajar** dan **Kelas**.

Kode Matakuliah	Matakuliah	Sks	Pengajar	Kelas
IF - 306	Sistem Informasi Manajemen	3	Paul	B-104
IF - 406	Basis Data	3	Anton	B-117
AK - 201	Mikroprosesor	4	Denny	H - 301
MK - 206	Struktur Data	4	Joko	C - 309

Tabel 4.7. Pengajar

3. Relasi **Nilai** dengan, atribut [**Nomor Mahasiswa** + **Kode Matakuliah**] sebagai kunci komposit dari atribut **Nilai**.

Nomor Mahasiswa	Kode Matakuliah	Nilai
92150001	IF - 306	A
92150001	IF - 406	B
92150002	IF - 406	A
92150002	AK - 201	B
92150002	MK - 206	B

Tabel 4.8. Relasi Nilai

Ketiga relasi diatas sudah merupakan normalisasi bentuk kedua, karena setiap atribut non-kuncinya sudah tergantung penuh terhadap kunci atributnya.

Tetapi pada relasi **Pengajar** masih terdapat anomali, yaitu :

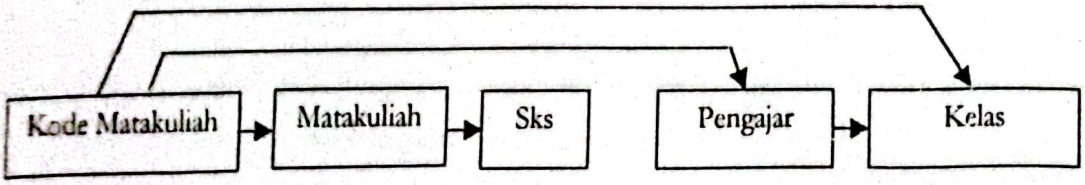
1. Bila **Kelas** dari **Pengajar** Anton pindah dari **B-117** ke **B-119**, maka harus dilakukan beberapa kali perubahan → anomali perubahan.

2. Bila ada **Matakuliah** yang akan dihapus, maka informasi mengenai **Pengajar** berikut **Kelas**-nya akan hilang → anomali penghapusan.

Sehingga harus dilakukan normalisasi bentuk ketiga. Sedangkan untuk relasi **Mahasiswa** dan **Nilai** sudah tidak terdapat lagi anomali dan ketergantungan transitif, maka relasi tersebut sudah *well-structured.*

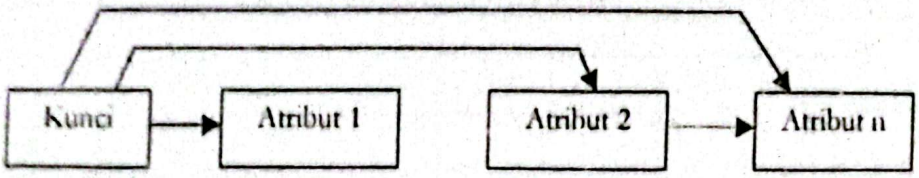
4.3. NORMALISASI BENTUK KETIGA (3NF)

Terjadinya anomali pada sebuah relasi **Pengajar,** yang sudah merupakan normalisasi bentuk kedua, disebabkan karena adanya *entity* tersembunyi pada relasi tersebut, yaitu *entity* **Pengajar**.



Gambar 4.2. Diagram Ketergantungan Pada Relasi Pengajar

Pada diagram diatas, setiap non-kunci atribut tergantung penuh terhadap kunci atributnya, tetapi ada non-kunci yang menjadi yang juga tergantung terhadap atribut non-kunci lainnya. Hal ini disebut ketergantungan transitif.



Gambar 4.3. Diagram Ketergantungan Transitif

Seperti halnya ketergantungan parsial, ketergantungan transitif akan mengakibatkan timbulnya anomali pada relasi yang bersangkutan. Kriteria bahwa relasi telah berada pada normalisasi bentuk ketiga (3NF), apabila pada relasi tersebut sudah tidak terdapat lagi ketergantungan transitif.

Untuk menghilangkan ketergantungan transitif pada relasi **Pengajar,** sehingga terbentuk normalisasi bentuk ketiga, maka relasi tersebut dipecah menjadi dua relasi yang baru, yaitu :

1. Relasi **Matakuliah – Pengajar** dengan atribut **Kode Matakuliah** sebagai atribut kunci **Matakuliah** dan **Pengajar.**
2. Relasi **Pengajar – Kelas** sebagai atribut kunci **Kelas.**

Kode Matakuliah	Matakuliah	Sks	Pengajar
IF - 306	Sistem Informasi Manajemen	3	Paul
IF - 406	Basis Data	3	Anton
AK - 201	Dasar - Dasar Akutansi	4	Denny
MK - 206	Pengantar Marketing	4	Joko

Tabel 4.9. Relasi MataKuliah - Pengajar

Pengajar	Kelas
Paul	B-104
Anton	B-117
Denny	H - 301
Joko	C - 309

Tabel 4.10. Relasi Pengajar - Kelas

Meskipun atribut **Pengajar** menjadi kunci pada relasi **Pengajar - Kelas**, tetapi relasi **Pengajar - Kelas** tetap merupakan atribut non-kunci pada relasi **Matakuliah - Pengajar**, maka atribut **Pengajar** disebut kunci tamu.

Sampai tahap ini proses normalisasi yang berkaitan dengan ketergantungan fungsional sudah selesai. Proses transformasi dari pandangan pengguna (*user-view*) berupa relasi yang belum dinormalisasikan sudah selesai menjadi empat buah bentuk normalisasi ketiga (3NF), yang sudah bersih dari anomali yang disebabkan oleh penyisipan, penghapusan dan perubahan.

Hal tersebut disebabkan oleh karena setiap *entity* disimpan pada relasinya masing – masing, sehingga data dapat disisipkan, dihapus dan dirubah tanpa memerlukan referensi, terutama *entity* lainnya.

4.4. NORMALISASI BENTUK BOYCE CODE (BCNF)

Pada umumnya relasi dalam bentuk normalisasi ketiga sudah memenuhi syarat untuk sebagian besar aplikasi basis data. Tetapi tidak berarti bahwa normalisasi bentuk ketiga sudah bebas dari anomali.

[illegible] langkah – langkah normalisasi, masih terdapat beberapa tahap normalisasi selain tiga adalah normalisasi biasa (*Boyce code normal form*)

[illegible] sebuah relasi memiliki lebih dari satu kunci kandidat, maka ada kemungkinan terdapat atribut pada relasi tersebut, meskipun relasi tersebut sudah pada tahap normalisasi bentuk ketiga.

Nama Mahasiswa	Matakuliah	Asisten
[illegible]	Sistem Informasi Manajemen	Nikki
[illegible]	Basis Data	Steve
[illegible]	Dasar - Dasar Akutansi	Doni
[illegible]	Pengantar Marketing	Irene

Tabel 4.11. Relasi Matakuliah Asisten

[illegible] semantik dari tabel diatas adalah :

1. Untuk setiap **Mahasiswa** boleh mengambil beberapa **Matakuliah**.
2. Untuk setiap **Matakuliah** hanya mempunyai seorang **Asisten**.
3. Untuk setiap **Asisten** hanya menjadi asisten pada satu **Matakuliah**.
4. Setiap **Asisten** dapat menjadi asisten dari beberapa **Mahasiswa** yang mengambil matakuliah.

Dari aturan semantik diatas, ketergantungannya dapat ditulis sebagai berikut :

Nomor Mahasiswa →→ Matakuliah

Nomor Mahasiswa, Matakuliah → Asisten

Matakuliah →→ Asisten

Asisten → Matakuliah

Asisten →→ Kode Matakuliah

Pada relasi **Matakuliah - Asisten**, tidak terdapat satupun atribut yang dapat dijadikan *determinant* dari dua atribut lainnya, sehingga tidak terdapat atribut tunggal yang dapat menjadi kunci kandidat. Tetapi terdapat dua kunci kandidat yang terbentuk dari kunci komposit, yaitu :

Nomor Mahasiswa, Asisten → Matakuliah

Nomor Mahasiswa, Matakuliah → Asisten

Bila dipilih **Nomor Mahasiswa, Matakuliah** sebagai kunci dari relasi **Matakuliah - Asisten,** maka relasi tersebut merupakan bentuk normal ketiga (3NF), karena didalamnya tidak terdapat ketergantungan parsial maupun ketergantungan transitif. Tetapi didalam relasi tersebut masih terdapat anomali, yaitu :

- Bila mahasiswa dengan nomor 92150001 batal mengambil matakuliah Sistem Informasi Manajemen, maka data Nikki sebagai asisten matakuliah tersebut juga akan hilang → anomali perubahan.
- Bila ada matakuliah baru, maka untuk memasukkan data matakuliah tersebut berikut asistennya harus menunggu sampai ada mahasiswa yang mengambil matakuliah tersebut → anomali perubahan.

- Bila Mahasiswa dengan Nomor Mahasiswa 92150001 berhenti kuliah, maka datanya harus dihapus, yang akan mengakibatkan informasi tentang asistennya akan ikut terhapus → anomali penghapusan.

Terdapatnya anomali pada relasi tersebut, karena terjadinya *overlapping candidate key*, yaitu kunci kandidat **Nomor Mahasiswa, Asisten** dan **Nomor Mahasiswa, Matakuliah** yang sama – sama menggunakan atribut **Nomor Mahasiswa**.

Untuk menghilangkan anomali tersebut, R.F. Boyce dan E.F. Codd, mengusulkan definsi yang lebih akurat untuk masalah diatas, yaitu definisi berdasarkan konsep dari sebuah determinan, dimana kriteria bahwa sebuah relasi menjadi BCNF, jika setiap determinan adalah kunci kandidat.

Relasi customer - customer category – sales, bisa dipecah menjadi dua relasi. Dimana atribut determinan yang bukan kunci kandidat dipisahkan pada relasi lain dan dijadikan kunci pada relasi tersebut. Ada dua bentuk pemisahan yang dapat dilakukan, yaitu :

1. Relasi **Mahasiswa – Asisten (Nomor Mahasiswa, Asisten)** dan **Asisten - Matakuliah (Asisten, Matakuliah).**
2. **Mahasiswa - Matakuliah (Nomor Mahasiswa, Matakuliah)** dan **Asisten – Matakuliah).**

Nomor Mahasiswa	Matakuliah
92150001	Sistem Informasi Manajemen
92150002	Basis Data
92150003	Basis Data
92150004	Dasar - Dasar Akutansi
92150005	Pengantar Marketing

Tabel 4.12 Relasi Mahasiswa – Asisten

Asisten	Matakuliah
Nikki	Sistem Informasi Manajemen
Steve	Basis Data
Doni	Dasar - Dasar Akutansi
Irene	Pengantar Marketing

Tabel 4.13 Relasi Asisten – Matakuliah

Nomor Mahasiswa	Matakuliah
92150001	Sistem Informasi Manajemen
92150002	Basis Data
92150003	Basis Data
92150004	Dasar - Dasar Akutansi
92150005	Pengantar Marketing

Tabel 4.14. Relasi Mahasiswa - Matakuliah

4.5. NORMALISASI BENTUK KEEMPAT (4NF)

Bila sebuah relasi sudah berada pada tahap BCNF, maka pada relasi tersebut tidak terdapat lagi anomali yang disebabkan oleh ketergantungan fungsional. Tetapi kemungkinan terdapatnya anomali pada relasi BCNF masih ada, yaitu anomali yang disebakan oleh ketergantungan banyak nilai.

Matakuliah	Pengajar	Buku Acuan
Basis Data	Doni	Korth and Silberschatz
	Andi	Date
	Karel	Mittra
Sistem Informasi Manajemen	Alex	McCleod, Jr
	Anton	Moekijat

Tabel 4.15. Relasi Matakuliah - Buku

Relasi diatas diatas mempunyai asumsi – asumsi sebagai berikut :

Matakuliah →→ Pengajar

Matakuliah →→ Buku Acuan

Pengajar ←/→ Buku Acuan

Bedasarkan ketiga asumsi diatas, dilakukan normalisasi terhadap relasi **Matakuliah – Buku**.

Matakuliah	Pengajar	Buku Acuan
Basis Data	Doni	Korth and Silberschatz
Basis Data	Doni	Date
Basis Data	Doni	Mittra
Basis Data	Andi	Korth and Silberschatz
Basis Data	Andi	Date
Basis Data	Andi	Mittra
Basis Data	Karel	Korth and Silberschatz
Basis Data	Karel	Date
Basis Data	Karel	Mittra
Sistem Informasi Manajemen	Alex	McCleod, Jr
Sistem Informasi Manajemen	Alex	Moekijat
Sistem Informasi Manajemen	Anton	McCleod, Jr
Sistem Informasi Manajemen	Anton	Moekijat

Tabel 4.16. Relasi Matakuliah – Buku Bentuk Normalisasi

Pada setiap **Matakuliah**, semua kombinasi antara struktur dengan **Buku Acuan** merupakan tupel – tupel pada relasi diatas.

Relasi diatas tidak terdapat determinan, maka kunci utamanya terdiri dari ketiga atribut diatas (**Matakuliah, Pengajar, Buku Acuan**). Karena tidak terdapat determinan selain kunci utama, maka relasi diatas sudah merupakan normalisasi boyce code (BCNF). Namun demikian masih terdapat *redundancy* data yang dapat mengakibatkan terjadinya anomali.

Bila akan ditambahkan sebuah buku acuan, akan menyebabkan bertambahnya tiga baris baru pada relasi tersebut untuk masing – masing pengajar. Hal tersebut disebabkan oleh ketergantungan banyak nilai.

Ketergantungan banyak nilai terjadi bila pada sebuah relasi paling tidak terdapat tiga buah atribut (misal A, B, C), dimana untuk setiap nilai A menjelaskan sejumlah nilai B dan sejumlah nilai C. Tetapi set nilai B terlepas dari set nilai C, begitu pula sebaliknya.

Untuk menghilangkan ketergantungan banyak nilai maka relasi **Matakuliah – Buku** tersebut memecah dua berdasarkan kemandirian datanya.

Matakuliah	Pengajar
Basis Data	Doni
Basis Data	Andi
Basis Data	Karel
Sistem Informasi Manajemen	Alex
Sistem Informasi Manajemen	Anton

Tabel 4.17. Relasi Matakuliah – Pengajar

Matakuliah	Buku Acuan
Basis Data	Korth and Silberschatz
Basis Data	Date
Basis Data	Mittra
Sistem Informasi Manajemen	McCleod, Jr
Sistem Informasi Manajemen	Moekijat

Tabel 4.18. Relasi Buku

Dari uraian diatas, maka dapat disimpulkan bahwa relasi bentuk keempat (4NF) merupakan relasi yang sudah merupakan bentuk normalisasi boyce code (BCNF) dan tidak terdapat lagi ketergantungan banyak nilai.

4.6. NORMALISASI BENTUK KELIMA (5NF)

Normalisasi tahap kelima ini dirancang untuk mengatasi jenis ketergantungan yang disebut ketergantungan join. Ketergantungan join terjadi pada relasi yang bila dipecah dua, maka relasi hasil pemisahan tersebut tidak dapat digabungkan kembali menjadi relasi semula.

BAB 5
IMPLEMENTASI DENGAN ORACLE POWER OBJECT

5.1. ORACLE

Oracle yang dibuat oleh Oracle Corporation of Belmont, California, merupakan perangkat lunak komersial pertama yang menggunakan bahasa *structured query language* untuk *relational data base management system* (RDBMS), artinya semua operasi terhadap terhadap basis data dilakukan dengan memberikan perintah SQL. Beberapa fasilitas yang diberikan memang terlihat seperti tidak menggunakan SQL, namun sebenarnya dibalik layar semua kemudahan yang diberikan fasilitas – fasilitas tersebut tetap dilaksanakan dengan SQL. Oracle juga memberikan fasilitas agar kita dapat bekerja secara interaktif dengan perintah SQL, seperti SQL*Plus pada Personal Oracle. Selain itu Personal Oracle juga mempunyai fasilitas PL/SQL, yang merupakan perintah percabangan (*condition*) dan perintah pengulangan (*looping*).

Berdasarkan lokasi penyimpanannya, basis data Oracle dapat dibagi atas dua bagian yaitu *remote database* dan *local database*.

- **Remote database** diartikan sebagai basis data yang ditempatkan kepada jaringan server (*back end*) yang terpisah dari aplikasi (*client*) dan dapat diakses oleh beberapa aplikasi yang berbeda – beda. Contoh dari *remote database* adalah **Oracle** dan **SQL Server.**
- **Local database** diartikan sebagai basis data yang berada pada tempat yang sama dengan aplikasi *front end / client* dan hanya dapat diakses pada aplikasi tersebut. Contoh lokal database adalah *blaze* ataupun Personal Oracle Lite yang terdapat pada Oracle Power Object.

5.2. ORACLE POWER OBJECT

Oracle Power Object adalah sebuah produk yang dirancang khusus untuk lingkungan *client / server*. Sebagai sebuah produk perangkat lunak aplikasi, Oracle Power Object terbilang istimewa karena memadukan antara kemudahan dengan kemampuan yang tinggi.

Kemudahan dalam pengoprasian Oracle Power Object tercermin dari penggunaan *graphical environment* dan fsilitas *drag and drop*, yang memungkinkan untuk membuat aplikasi yang sederhana tanpa penulisan program (*code*) sama sekali.

Oracle Power Object dikembangkan oleh Oracle Inc., dan berjalan pada sistem operasi Windows dan Macintosh. Oracle Power Object memiliki *tools*, *methods* dan *controls* yang lengkap untuk membuat, mengubah dan memelihara komponen – komponen aplikasi seperti *application*, *form*, dan *report*.

Oracle Power Object dirancang agar pengembang dapat menggunakan *object oriented programming* dalam membuat aplikasi. Penggunaan teknik ini memberikan keleluasaan kepada pengembang untuk membuat *classes* yang dapat digunakan secara bersama oleh beberapa aplikasi. Cara ini dapat menghemat waktu, karena *object* yang sama hanya perlu dibuat sekali di dalam *classes* selanjutnya, bisa digunakan secara bersamaan.

Beberapa kemampuan dasar dari Oracle Power Object, antara lain :

- **Graphical Design Tools**. Oracle Power Object menyediakan banyak *form tools* dan *report tools* yang dapat dipergunakan untuk merancang *interface* sesuai dengan kebutuhan *user*.
- **Object Oriented Design Future**. Oracle Power Object dirancang dengan menggunakan pendekatan *objects* yang meliputi *classes*, *properties* dan *methods*.

Pendekatan *object* ini memungkinkan *user* untuk merancang berbagai komponen dasar untuk mengembangkan aplikasi basis data.

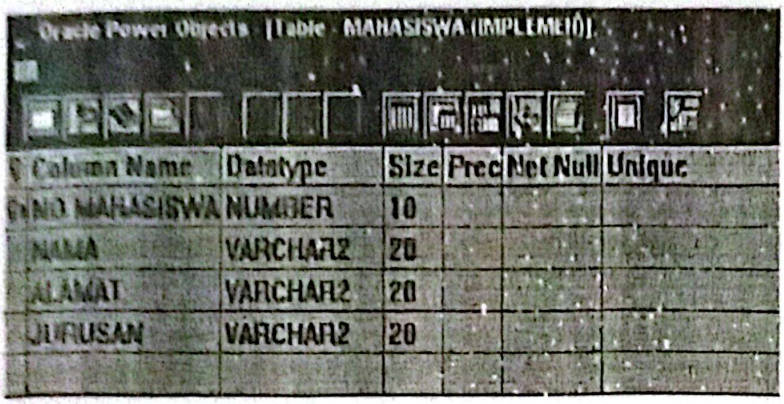
- **A Complete Programming Language**. Oracle Power Object menggunakan Oracle basic sebagai bahasa pemrograman yang relatif mudah untuk dipelajari.
- **Included Local Database**. Oracle Power Object memberikan basis data *blaze*, sebuah relasional basis data yang dirancang khusus untuk Oracle Power Object. Dengan *blaze*, user dapat membangun sebuah *local database* yang sesuai untuk aplikasi sederhana.
- **Support for Leading Relational Database System.** Oracle Power Object dirancang untuk dapat mengakses berbagai sumber basis data dengan cara yang sama, termasuk *Oracle Server* dan *Microsoft SQL Server.*

Jenis – jenis file yang terdapat di Oracle Power Object adalah :

- **BLZ**, merupakan *extension file* untuk *database blaze.*
- **ODB**, merupakan *extension file* untuk *database personal oracle lite.*
- **POS**, merupakan *extension file* untuk *session.*
- **POL**, merupakan *extension file* untuk *library.*
- **POA**, merupakan *extension file* untuk *application.*
- **PO**, merupakan *extension file* untuk *compile run.* Jenis aplikasi ini biasanya berukuran kecil dan tidak dapat berdir sendiri, dalam menjalankan jenis aplikasi ini dibutuhkan Oracle Power Object *run time.*
- **EXE**, merupakan *extension file* untuk *compile standalone application.* Kebalikan dari PO, jenis ini mempunyai ukuran yang besar dan dapat langsung pada *Windows.*

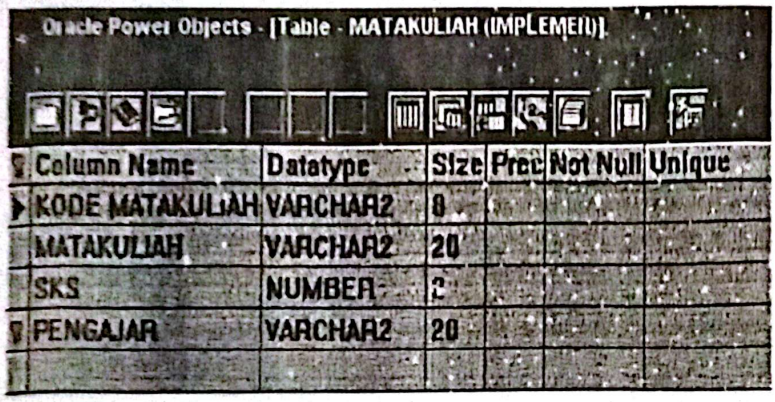
Untuk membuat struktur tabel dalam Oracle Power Object, harus memiliki kondisi – kondisi yang harus dipenuhi, yaitu adanya basis data yang menampung basis data (*blaze*) dan harus ada koneksi antara Oracle Power Object dengan basis data melalui sesi (*session*).

Dari tabel – tabel hasil normalisasi, dibentuk struktur tabel sebagai berikut :


Oracle Power Objects - [Table - MAHASISWA (IMPLEMEN)]

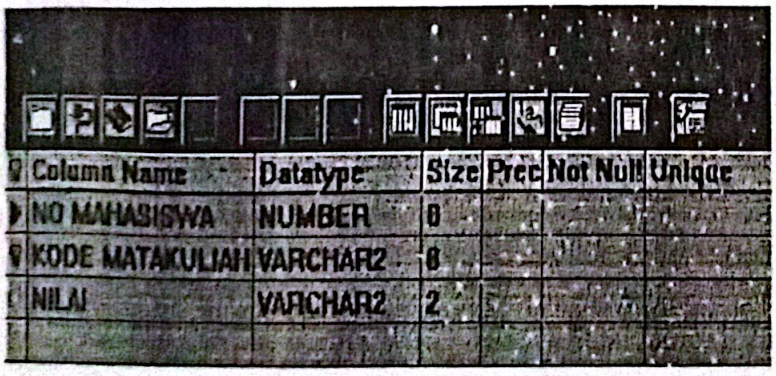
Column Name	Datatype	Size	Prec	Not Null	Unique
NO MAHASISWA	NUMBER	10			
NAMA	VARCHAR2	20			
ALAMAT	VARCHAR2	20			
JURUSAN	VARCHAR2	20			

Gambar 5.1. Struktur Tabel Mahasiswa


Oracle Power Objects - [Table - MATAKULIAH (IMPLEMEN)]

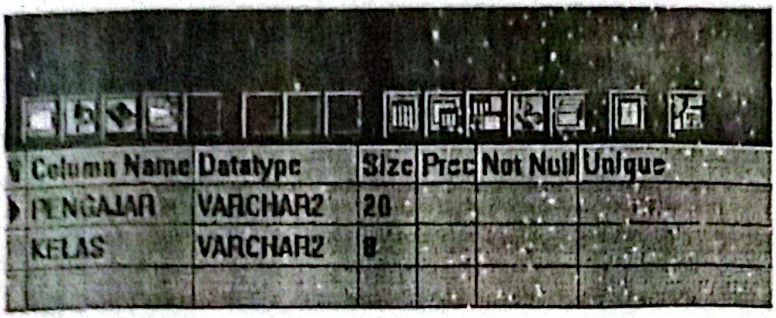
Column Name	Datatype	Size	Prec	Not Null	Unique
KODE MATAKULIAH	VARCHAR2	8			
MATAKULIAH	VARCHAR2	20			
SKS	NUMBER	2			
PENGAJAR	VARCHAR2	20			

Gambar 5.2. Struktur Tabel Matakuliah - Pengajar



Column Name	Datatype	Size	Prec	Not Null	Unique
NO MAHASISWA	NUMBER	8			
KODE MATAKULIAH	VARCHAR2	8			
NILAI	VARCHAR2	2			

Gambar 5.3. Struktur Tabel Nilai



Column Name	Datatype	Size	Prec	Not Null	Unique
PENGAJAR	VARCHAR2	20			
KELAS	VARCHAR2	8			

Gambar 5.4. Struktur Tabel Pengajar - Kelas

Beberapa hal yang harus diketahui dalam pembuatan tabel seperti :

Komponen Tabel

Nama Kolom	**Keterangan**
Datatype	Tipe data yang dapat diinputkan kedalam tabel
Size	Maksimum karakter yang dapat dimasukkan kedalam kolom suatu tabel
Precision	Jumlah digit angka dibelakang koma
Not Null	Berarti kolom tersebut harus mempunyai nilai
Unique	Semua nilai didalam kolom tidak ada duplikasi

Tipe Data

Tipe Data	Keterangan
Varchar2	Karakter string, tidak berdasarkan maksimum karakter saat definisi tabel
Char	Karakter string, berdasarkan maksimum karakter saat definisi tabel
Date	Tanggal
Number	Angka numerik antara –2.147.483.648 s/d +2.147.483.648 dan bukan angka desimal
Integer	Angka Numerik antara –32.768 s/d + 32.768 dan bukan angka desimal
Float	Angka desimal sampai dengan 15 digit
Long	Menyimpan tipe data *binary*
Raw / Long Raw	Menyimpan tipe data *binary*

BAB 6
KESIMPULAN

Proses normalisasi adalah proses mendekomposisikan tabel – tabel menjadi beberapa tabel kecil yang simpel dalam *field – filed*-nya dan benar menurut struktur dekomposisi.

Penggunaan normalisasi pada basis data dapat :

- Menghilangkan duplikasi data dalam suatu tabel.
- Meminimumkan ketidakkonsistenan data dalam basis data. Setelah suatu tabel dilakukan proses normalisasi maka data yang disisipkan dalam suatu tabel tidak akan mempengaruhi proses manipulasi data, data yang dihapus dari tabel tidak akan mengurangi informasi yang terkandung didalam tabel dan perubahan data dapat dilakukan lebih efisien.
- Proses manipulasi data seperti *indexing*, *searching / selecting*, *updating* dan pemeliharaan integritas data menjadi lebih mudah dan cepat.
- Dekomposisi tabel – tabel data akibat normalisasi dapat dibentuk kembali menjadi tabel asal dengan operasi join tanpa kehilangan informasi akibat penggabungan tebel – tebel hasil normalisasi.
- Karena tabel ditampilkan dalam jumlah atribut yang lebih sedikit, maka tabel akan lebih mudah dibaca.

Tetapi penggunaan metode normalisasi menyebabkan media penyimpanan data menjadi sangat besar, karena beberapa *filed* disertakan kembali kedalam tabel – tebel hasil dekomposisi.

DAFTAR PUSTAKA

- **[Korth and Silbereshatz 1991]** Henry F. Korth and Abraham Silberschatz, "Database System Concept," McGraw-Hill Book Co, Singapura (1991).
- **[Mittra 1991]** Sitansu S. Mittra, "Principles of Relational Database System," Prentice-Hall International, New Jersey (1991).
- **[Date]** C. J. Date, "An Introduction to Database Systems," Addison-Wesley Publishing Company, Inc, USA (1995).
- **[McCleod, Jr]** Raymond McCleod, Jr, "Management Information System, *A Study of Computer-Based Information Systems*," Prentice-Hall, Inc, New Jersey (1995).
- **[Abbey and Corey]** Michael Abbey and Michael J. Corey, "ORACLE, *Beginner's Guide*," Osborne McGraw-Hills, California (1995).
- "Oracle Power Object™, *Users Guide*", Oracle Corporation (1995)
- "Oracle Power Object™, *Getting Started*", Oracle Corporation (1995)
- "Oracle Power Object™, *Sample Applications*", Oracle Corporation (1995)