Materi 4

BASIS DATA

3 SKS | Semester 2 | S1 Sistem Informasi | UNIKOM | 2017

Nizar Rabbi Radliya | nizar@email.unikom.ac.id

Nama Mahasiswa	
NIM	
Kelas	

Kompetensi Dasar

Memahami konsep dasar normalisasi data dalam perancangan basis data.

Pokok Bahasan

Pengantar Normalisasi Data
1. Pengenalan 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 funsional
   b. Dependensi sepenuhnya
   c. Dependensi parsial
   d. Dependensi total
   e. Dependensi transitif

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 manipulasi (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

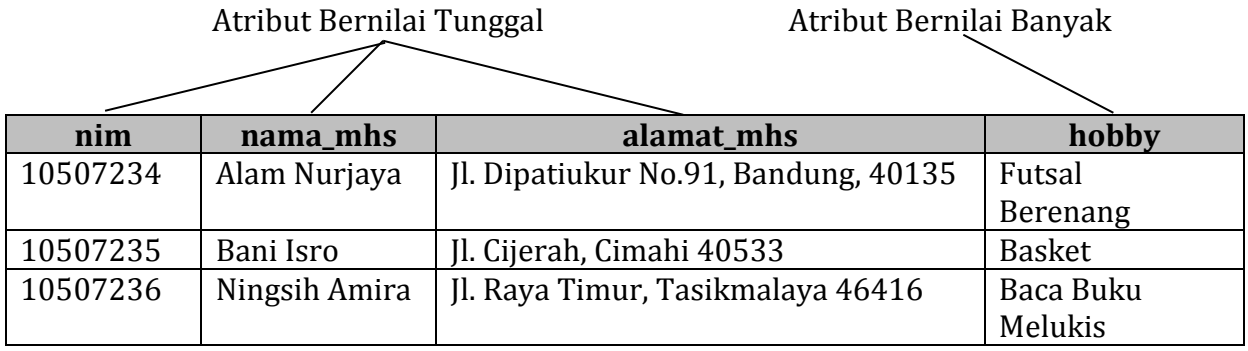
Alamat	kota	kode_pos
Jl. Dipatiukur No.91	Bandung	40135
Jl. Cijerah No.20	Cimahi	40533
Jl. Raya Timur No.321	Tasikmalaya	46416

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.



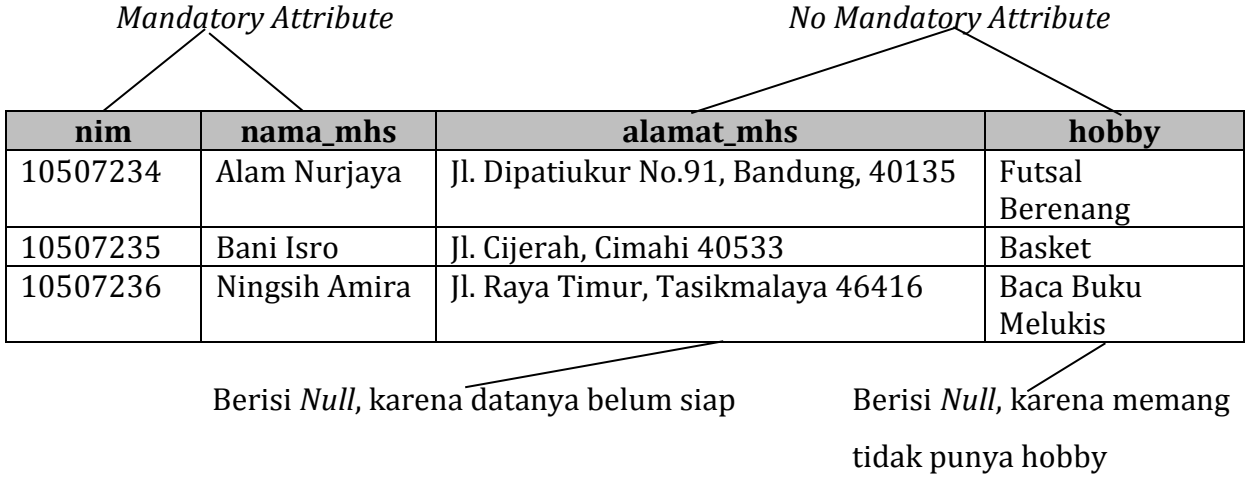
nim	nama_mhs	alamat_mhs	hobby
10507234	Alam Nurjaya	Jl. Dipatiukur No.91, Bandung, 40135	Futsal Berenang
10507235	Bani Isro	Jl. Cijerah, Cimahi 40533	Basket
10507236	Ningsih Amira	Jl. Raya Timur, Tasikmalaya 46416	Baca Buku Melukis

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* merupkan 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.



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 3. *Mandatory Attribute* dan Nilai Null

Nilai *Null* tidak sama dengan spasi, walaupun pada waktu nilai ditampilkan sama-sama tidak memperilihatkan 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 ekivalen dengan karakter ke-32 dalam tabel ASCII, sedangkan nilai *Null* ekivalen 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 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 dijit 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 sebuat 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 terkati 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 dimasukan 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 maslah 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 minimbulkan 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 pendekomposisian data kedalam bentuk yang paling efisien. Ada beberapa jenis dependesi 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 dependesi fungsional terhadap atribut X jika dan

hanya jika setiap nilai X berhubungan dengan sebuah nilai Y. Dependensi fungsional Y terhadp X dapat dinotasikan sebagai berikut:

X -> 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:

kode_barang -> 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} -> Z

Notasi di atas menyatakan bahwa atribut Z mempunya dependensi fungsional terhadap paasangan 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:

{no_dosen, pendidikan} -> tahun_lulus

Dimana tidak setiap *no_dosen* menetukan *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} -> tahun_lulus

Dimana tidak setiap *no_dosen* menetukan *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 -> nama_pemasok

nama_pemasok -> kode_pemasok

Dengan demikian berlaku penotasian sebagai berikut:

kode_pemasok <-> nama_pemasok

Sebuah nilai *kode_pemasok* hanya akan berpasangan dengan sebuah *nama_pemasok*, dan begitu juga sebaliknya. Hal tersebut dengan asumsi tidak aka nada 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 -> Y -> Z

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

kode_barang -> kode_pemasok -> 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. 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?

V. Materi Berikutnya

Pokok Bahasan	Tahapan Normalisasi Data
Sub Pokok Bahasan	1. Bentuk tidak normal 2. Bentuk normal pertama 3. Bentuk normal kedua 4. Bentuk normal ketiga 5. Bentuk normal boyce-codd 6. Bentuk normal keempat 7. Bentuk normal kelima

VI. Daftar Pustaka

Fathansyah. 2012. Basis Data. Bandung: Informatika.
Kadir, A. 2009. Dasar Perancangan dan Implementasi Database Relasional. Yogyakarta: Andi.
Kristanto, H. 2004. Konsep dan Perancangan Database. Yogyakarta: Andi.
Nugroho, A. 2004. Konsep Pengembangan Sistem Basis Data. Bandung: Informatika.
Nugroho, B. 2005. Database Relasional dengan MySQL. Yogyakarta: Andi.
Simarmata, J. 2007. Perancangan Basis Data. Yogyakarta: Andi.