BAB 4
NORMALISASI DATA

Perancangan basis data diperlukan, agar kita bisa memiliki basis data yang kompak dan efisien dalam penggunaan ruang penyimpanan, cepat dalam pengaksesan dan mudah dalam pemanipulasian (tambah, ubah, hapus) data.

Dalam merancang basis data, kita dapat melakukannya dengan :

1. Menerapkan normalisasi pada struktur *table* yang telah diketahui.
2. Langsung membuat model ER (*Entity Relationship*)

Proses **Normalisasi,** merupakan proses pengelompokan data elemen menjadi *table-table* yang menunjukkan *entity* dan relasinya. Pada proses normalisasi selalu diuji pada beberapa kondisi, apakah ada kesulitan pada saat menambah / *insert*, menghapus / *delete*, mengubah / update, dan membaca / *retrieve* pada suatu *Database.*

Bila ada kesulitan pada pengujian tersebut, maka relasi tersebut dipecahkan menjadi beberapa *table* lagi, sehingga diperoleh *database* yang optimal.

4.1. *Atribut Table*

Normalisasi lebih difokuskan pada tinjauan komprehensif terhadap setiap kelompok data (*table*) secara individual. Lebih jauh tinjauan tersebut dititikberatkan pada data di masing – masing kolom pembentuk tabel. Kita menggunakan istilah baru, yaitu atribut yang sebenarnya identik dengan pemakaian istilah kolom data atau *field.*

Sebagai contoh :

- *Table* Dosen, memiliki 9 buah field / atribut, yaitu : nid, nama_d, tempat_lhr, tgl_lahir, jkelamin, alamat, kota, kodepos, dan gajipokok.
- *Table* Matakuliah, memiliki 4 buah atribut, yaitu : kdmk, nama_mk, sks dan semester.
- *Table* Jurusan, memiliki 4 buah atribut, yaitu : kode_jur, nama_jur, jenjang, dan nama_kajur.

- *Table* Mengajar, memiliki 9 buah atribut, yaitu : nid, thn_akademik, smt, hari, jam_ke, kdmk, waktu, kelas, dan kode_jur.

Dari 4 *table* di atas, masing-masing *table* memiliki *field / atribut* sebagai *key*, dan lainnya sebagai *atribut deskriptif*. Ada pula atribut yang tergolong atribut sederhana atau komposit dan lainnya.

4.1.1. *Key* dan *Atribut Deskriptif*

Key, adalah satu atau gabungan beberapa atribut yang dapat membedakan semua baris data (row) dalam tabel secara unik. Artinya adalah apabila suatu *field / atribut* dijadikan *key*, maka tidak boleh ada dua atau lebih baris data dengan nilai yang sama untuk *field / atribut* tersebut.

Sehubungan dengan pernyataan tersebut, maka kita dapat membedakan 3 (tiga) macam *key* yang dapat diterapkan pada suatu tabel :

1. *Superkey.*
2. *Candidate-Key.*
3. *Key Primer (Primary-Key).*
4. Kunci tamu (*Foreign Key)*

a. Superkey

Superkey adalah merupakan satu atau lebih *field / atribut* (kumpulan atribut) yang dapat membedakan setiap baris data dalam sebuah *table* secara unik. Bisa terjadi, ada lebih dari 1 kumpulan atribut yang bersifat seperti itu pada sebuah *table.*

Pada tabel Dosen yang memiliki 4 buah *field / atribut* tersebut, yang dapat menjadi *superkey adalah :*

- (nid, nama_d, tempat_lhr, tgl_lahir, jkelamin, alamat, kota, kodepos, gajipokok)
- (nid, nama_d, tempat_lhr, tgl_lahir, jkelamin, alamat, kota, kodepos)
- (nid, nama_d, tempat_lhr, tgl_lahir, jkelamin, alamat, kota,)
- (nid, nama_d, tempat_lhr, tgl_lahir, jkelamin, alamat,)
- (nid, nama_d, tempat_lhr, tgl_lahir, jkelamin,)
- (nid, nama_d, tempat_lhr, tgl_lahir)
- (nid, nama_d, tempat_lhr)
- (nid, nama_d,)
- (nid)

- (nama_d)

b. Candidate-Key

Candidate_key adalah merupakan kumpulan *field / atribut* minimal yang dapat membedakan setiap baris data dalam sebuah *table* secara unik.Sebuah *Candidate-key* tidak boleh berisi atribut atau kumpulan atribut yang telah menjadi superkey yang lain. Jadi sebuah *Candidate-key* pasti *superkey*, tapi belum tentu sebaliknya.

Pada *table* Dosen tersebut diatas, yang dapat menjadi *Candidate-key* adalah :

- (nid).
- (nama_d), jika kita bisa menjamin tidak ada nilai yang sama untuk atribut ini.

Pada sebuah *table* dimungkinkan ada lebih dari satu *Candidate-key*, seperti contoh diatas. Salah satu dari *Candidate–key* ini (jika memang lebih dari satu) dapat dijadikan sebagai *Key Primer (Primary key).*

c. Primarye-Key

Primary_key adalah candidate-key yang dipilih untuk mengidentifikasi tupel secara unik pada suatu relasi. Kunci utama dapat terbentuk dari satu atribut atau lebih.

Pemilihan *Key* Primer dari sejumlah *Candidate-key* pada suatu *table* didasari pada ketiga hal berikut ini:

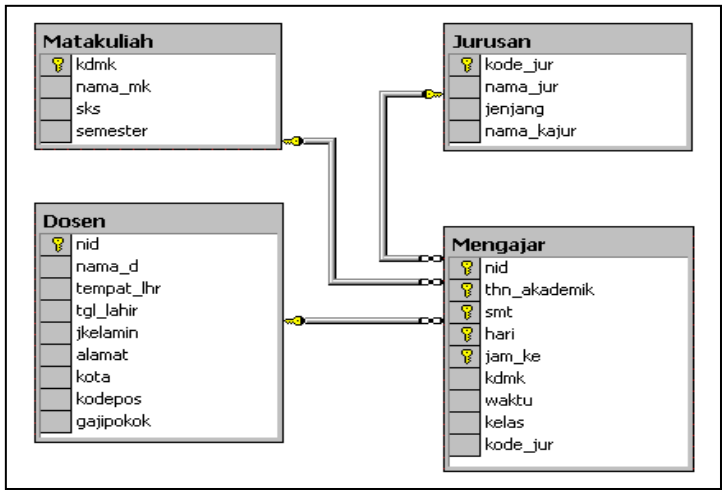
1. *Key* tersebut lebih sering (natural) untuk dijadikan acuan.
2. *Key* tersebut lebih ringkas.
3. Jaminan keunikan *Key* tersebut lebih baik.

Dengan pertimbangan tersebut, kedua *Candidate-key* pada *table* dosen, yaitu nid dan nama_d, yang lebih cocok sebagai ***Key Primer*** adalah (nid). Hal ini dikarenakan bahwa jaminan keunikan daripada nid (nomor induk dosen) akan terjamin karena karena setiap nid pada suatu perguruan tinggi pastinya tidak akan sama nilainya, sedangkan kenapa tidak memilih nama_d, karena nama_d kemungkinan ada yang sama nilainya.

c. Foreign-Key

Kunci tamu adalah satu atribut (satu set atribut) yang melengkapi satu *relationship* (hubungan) yang menunjukkan ke induknya. Kunci tamu ditempatkan pada relasi anak dan sama dengan kunci primer induk direlasikan.

Hubungan antara relasi induk dengan anak adalah **satu lawan banyak** (*one to many relationship*).



Gambar 4.1. Relasi *one to many*

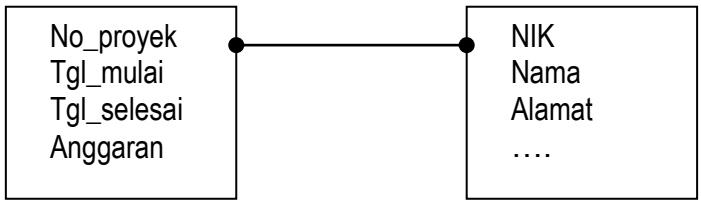
Kunci tamu dari *table* / relasi mengajar adalah : nid, kdmk dan kode_jur karena dalam *table* ini membutuhkan data – data yang ada dalam *table* dosen, matakuliah dan jurusan

Kunci primer untuk table / relasi mengajar adalah nid,thn_akademik,smt,hari,jam_kei, karena unik dan mewakili entity.

Dalam hal hubungan 2 buah relasi yang mempunyai *relation* banyak ke banyak (*many to many*), maka terdapat 2 buah kunci tamu pada relasi konektornya.

Contoh:

Relasi Proyek	RelasiPegawai

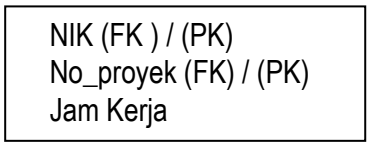


Gambar 4.2. Relasi *many to many*

Hubungan relasi proyek dengan pegawai adalah *many to many*, dengan pengertian bahwa satu pegawai mengerjakan lebih dari 1 proyek dan 1 proyek dikerjakan oleh beberapa pegawai.

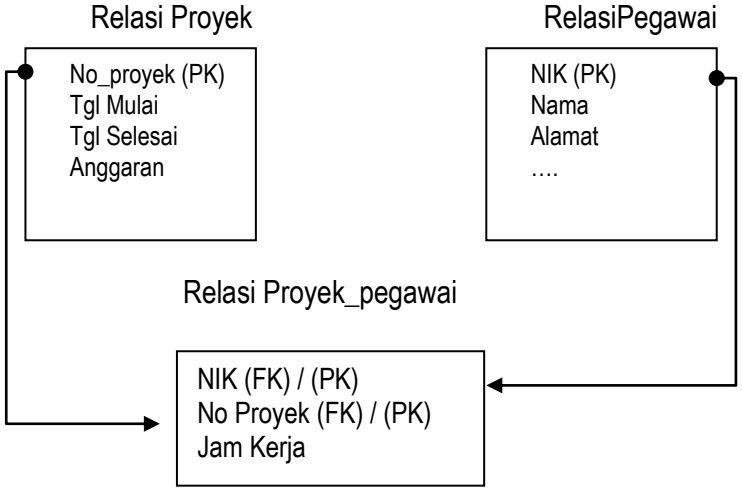
Untuk menunjukkan hubungan tersebut, maka dipakai relasi konektor yang berisi kunci tamu dari kedua relasi (proyek dan pegawai). Sehingga relasi konektor proyek_pegawai berisi atribut :

Relasi Proyek_pegawai



Pada relasi proyek_pegawai tersebut atribut nik dan no_proyek merupakan kunci tamu (FK) dan keudnya juga menjadi *primay key*, dan keduanya merupakan kunci primer (*primary key*) pada relasi induknya.

Sehingga hasilnya menjadi sebagai berikut :



Keterangan : FK = Foreign Key (kunci tamu), PK = Primary Key (kunci utama)

Gambar 4.3. Dekomposisi many to many

d. Atribut Deskriptif

Atribut diskriptif adalah atribut – atribut yang tidak menjadi atau merupakan anggota dari *Key Primer* pada suatu *table* didalam database. Pada *table* dosen diatas, yang menjadi atribut diskriptif adalah nama_d, tempat_lhr, tgl_lahir, jkelamin, alamat, kota, kodepos, gajipokok.

4.1.2. Atribut Sederhana dan Atribut Komposit

Atribut sederhana adalah merupakan atribut atomik yang tidak dapat dipilah lagi menjadi lainnya, sedangkan a*tribut komposit* adalah merupakan atribut yang masih dapat diuraikan lagi menjadi sub-sub atribut yang masing-masing memiliki makna.

Sebagai contoh pada *table* dosen tersebut diatas:

- Nama_d, tempat_lhr merupakan contoh atribut sederhana, karena tidak dapat diuaraikan lagi menjadi atribut lainnya.
- Alamat seorang dosen adalah merupakan contoh atribut komposit, karena dapat diuraikan menjadi beberapa sub atribut seperti : alamat_jln, kelurahan,kecamatan rt,rw,no_rumah, yang masing – masing memiliki makna tersendiri.

Contoh Atribut sederhana:

nama_d	tempat_lhr
Riswoko Sasono,MMSi	Bogor
Hasta Riyanti,Ir,MM	Jakarta
Cokro Diningrat,Ir,MT	Bekasi
Sakib Aljaber,MT	Cikarang
Bambang Sutedjo,Ir,MMSi	Jakarta
Asri Kasetyaningsih,M.Kom	Semarang
Triyatno,Ir,MM,M.Kom	Bekasi
Diastuti Pujiningsih,MM,MT	Surabaya
Endang Junianti,Ir,MMSi	Bandung
Djoko Pamungkas,M.Kom	Bogor
Didik Atmadja,Ir,MMSi	Bandung
Bagus Windarjo,M.Kom.	Tangerang
Dewi Anjani,Ir,MM	Bekasi

Contoh Atribut Komposit:

alamat
Jl. Catur 3 No.23
Jl. Delima 2 No.2
Jl. Dayang Raya 2 No.12
Jl. Kemuning 1 No.1
Jl. Mawar 6 No.2
Jl. Perjuangan 3 No.11
Jl. Mawar Indah 1 No.1
Jl. Janur Kuning 2 No.56
Jl. Kemukus 2 No.56
Jl. Anggrek 7 No.1
Jl. Cipete Raya No.6
Jl. H. Ali No.21
Jl.Kemang Raya 2 No.1

Gambar 4.4. Atribut sederhana dan komposit

Pada atribut alamat, dipecah menjadi 6 (enam) atribut (alamat_jln, kelurahan,kecataman,rt,rw,no_rumah), **bertujuan** jika kelak ada rencana / kebutuhan untuk melakukan pengolahan data terhadap sub-sub atribut tersebut. Misalkan untuk mencari dosen yang tinggal di kelurahan atau kecamatan tertentu siapa saja, atau ingin mecari dosenj yang tinggal di kota tertentu (Bekasi misalnya) siapa saja dapat dihasilkan dengan mudah dengan memanfaatkan fasilitas SQL (*query*).

4.1.3. Atribut Bernilai Tunggal dan Atribut Bernilai Banyak

Atribut bernilai tunggal ditujukan pada atribut – atribut yang memiliki paling banyak satu nilai untuk setiap baris data.

Pada *table* dosen, atribut (nid, nama_d, tempat_lhr, tgl_lahir, jkelamin, alamat, kota, kodepos, gajipokok) merupakan atribut bernilai tunggal, karena atribut-atribut tersebut hanya dapat berisi satu nilai. Jika ada seorang dosen yang memiliki 2 tempat tinggal, maka hanya salah satu saja yang boleh dimasukkan ke atribut alamat.

Atribut bernilai banyak ditujukan pada atribut-atribut yang dapat kita isi dengan lebih dari satu nilai, tetapi jenisnya sama. Pada *table* dosen, kita dapat menambahkan atribut hobbi. Seorang dosen ada yang mempunyai 1 hobi, banyak hobi, bahkan ada yang tidak punya hobi.

Atribut bernilai **Tunggal**

Atribut bernilai **Banyak** (hobi)

nid	nama_d	tempat_lhr	tgl_lahir	jkelamin	alamat	kota	kodepos	gajipokok	hobi
00001	Riswoko Sasono,MMSi	Bogor	12/24/74	Pria	Jl. Catur 3 No.23	Bekasi Utara	52123	1300000	Membaca, Renang
00002	Hasta Riyanti,Ir,MM	Jakarta	5/27/69	Wanita	Jl. Delima 2 No.2	Bekasi Timur	45612	1150000	Sepak Bola
01001	Cokro Diningrat,Ir,MT	Bekasi	12/19/66	Pria	Jl. Dayang Raya 2 No.12	Cibitung	54456	1200000	Tidur
01002	Sakib Aljaber,MT	Cikarang	5/20/70	Pria	Jl. Kemuning 1 No.1	Cikarang	56234	1150000	<NULL>
95001	Bambang Sutedjo,Ir,MMSi	Jakarta	2/23/58	Pria	Jl. Mawar 6 No.2	Jakarta Selatan	23234	1300000	Jalan - jalan
95002	Asri Kasetyaningsih,M.Kom	Semarang	12/25/62	Wanita	Jl. Perjuangan 3 No.11	Bekasi Timur	54567	1200000	<NULL>
96001	Triyatno,Ir,MM,M.Kom	Bekasi	5/14/67	Pria	Jl. Mawar Indah 1 No.1	Bekasi Barat	54356	1100000	<NULL>
96002	Diastuti Pujiningsih,MM,MT	Surabaya	12/24/70	Wanita	Jl. Janur Kuning 2 No.56	Cibitung	54566	1000000	<NULL>
97001	Endang Junianti,Ir,MMSi	Bandung	5/20/67	Wanita	Jl. Kemukus 2 No.56	Cikarang	56555	1150000	Tenis
97002	Djoko Pamungkas,M.Kom	Bogor	5/28/71	Pria	Jl. Anggrek 7 No.1	Bekasi Timur	45666	1100000	Bela Diri, Renang
98001	Didik Atmadja,Ir,MMSi	Bandung	4/20/70	Pria	Jl. Cipete Raya No.6	Jakarta Selatan	24123	1250000	Sepak Bola
98002	Bagus Windarjo,M.Kom.	Tangerang	12/13/71	Pria	Jl. H. Ali No.21	Cibitung	54523	1150000	<NULL>
99001	Dewi Anjani,Ir,MM	Bekasi	12/14/75	Wanita	Jl.Kemang Raya 2 No.1	Bekasi Barat	54567	1200000	Bulu Tangkis

Gambar 4.5. Atribut bernilai tunggal dan banyak

4.1.4. Atribut Harus Bernilai (Mandatory Attribute) dan Nilai Null

Mandatory Attribute adalah merupakan sejumlah atribut yang ada pada suatu *table* yang harus berisi data dan tidak boleh kosong.

Sedangkan Non Mandatory Attribute adalah sejumlah atribut yang ada pada suatu table yang boleh tidak diisi datanya / boleh kosong. *Nilai Null* digunakan untuk mengisi atribut – atribut yang nilainya memang belum siap / tidak ada. Misalkan pada table dosen kita tambahkan 1 (satu) record, seperti tampak pada gambar berikut:

98002	Bagus Windarjo,M.Kom.	Tangerang	12/13/71	Pria	Jl. H. Ali No.21	Cibitung	54523	1150000	<NULL>
99001	Dewi Anjani,Ir,MM	Bekasi	12/14/75	Wanita	Jl.Kemang Raya 2 No.1	Bekasi Barat	54567	1200000	Bulu Tangkis
99002	Bagus Tenan,M.Kom.	<NULL>	<NULL>	Pria	<NULL>	<NULL>	<NULL>	<NULL>	<NULL>

Gambar 4.6. Atribut harus bernilai dan nilai null

- Atribut nid dan nama_d adalah *atribut mandatory* dan nilainya tidak boleh kosong (*not null*), hal ini juga berhubungan pada saat kita mendesain *table* pada *database*, pada saat *table* dosen tersebut di *create* maka pendefinisian atribut tersebut harus *not null/* dan biasanya yang menjadi *primary key* pada suatu *table* pasti nilainya *not null.*
- Untuk atribut yang berisi null adalah karena data yang ada maasih meragukan atau belum ada sama sekali, jadi pada saat kita meng *create table* untuk atribut tersebut harus didefinisikan *null*, kalau kita mendefinisikan *not null*, walaupun datanya belum ada atau meragukan maka kita harus mengisi datanya.

Pada contoh diatas yang menjadikan *mandatory atribut* adalah nid dan nama_d,karena pada saat petugas melakukan input (pemasukan data) nid pasti ada nama_d, karena nim akan terbentuk kalau nama_d (nama dosen) nya ada, tidak mungkin ada nim sedangkan nama_d nya tidak ada dan pada atribut ini nilainya harus ada tidak boleh kosong (null) jadi harus isi datanya. Sedangkan untuk

atribut *non mandatory* adalah tempat_lhr,tgl_lahir,jkelamin,alamat dan lainnya, karena pada atribut tersebut nilainya boleh kosong dikarenakan datanya belum siap atau masih meragukan.

4.1.5. Atribut Turunan (*Derived Attribute*)

Atribut turunan adalah atribut yang nilai-nilainya diperoleh dari pengolahan atau dapat diturunkan dari atribut atau *table* lain yang berhubungan. Penambahan atribut tahun_masuk pada *table* dosen merupakan contoh atribut turunan.

nid	nama_d	tempat_lhr	tgl_lahir	jkelamin	alamat	kota	kodepos	gajipokok	tahun_masuk
00001	Riswoko Sasono,MMSi	Bogor	12/24/74	Pria	Jl. Catur 3 No.23	Bekasi Utara	52123	1300000	2000
00002	Hasta Riyanti,Ir,MM	Jakarta	5/27/69	Wanita	Jl. Delima 2 No.2	Bekasi Timur	45612	1150000	2000
01001	Cokro Diningrat,Ir,MT	Bekasi	12/19/66	Pria	Jl. Dayang Raya 2 No.12	Cibitung	54456	1200000	2001
01002	Sakib Aljaber,MT	Cikarang	5/20/70	Pria	Jl. Kemuning 1 No.1	Cikarang	56234	1150000	2001

Gambar 4.7. Atribut turunan

Atribut tahun_masuk pada contoh diatas dapat ditiadakan, karena atribut tersebut dapat diperoleh dari atribut nid, yaitu 2 digit pertama yang menyatakan tahun masuk dosen, jadi tidak perlu dibuat untuk atribut tersebut.

Pertanyaan Soal

1. Jelaskan pengertian daripada *Key* dan atribut deskriptif ?.
2. Jelaskan dan berikan contoh pengertian dari *superkey, candidate key, primary key dan foreign key* ?.
3. Jelaskan dan berikan contoh pengertian daripada atribut sederhana dan komposit ?.
4. Jelaskan dan berikan contoh pengertian daripada atribut bernilai tunggal dan bernilai banyak ?.
5. Jelaskan dan berikan contoh pengertian daripada atribut turunan ?.
6. Berikan penjelasan, mana yang merupakan *superkey, candidate key*, *primary key dan foreign key* dari *table (relasi)* Matakuliah, Jurusan dan Mengajar dari *database* Penjadwalan_mengajar_dosen ?.