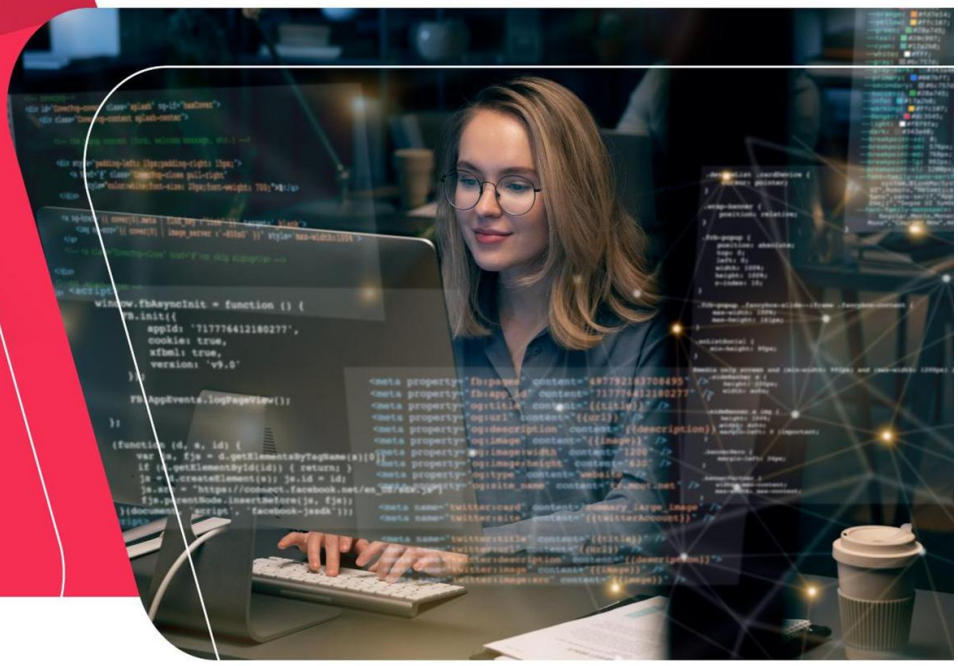
ANALISIS DAN DESIGN
SISTEM MANAJEMEN BASIS DATA
DAN
STRUCTURED QUERY LANGUAGE

TIA-Academy



www.ti-asia.com

Agus Sasmito Aribowo, S.Kom, M.CS

BAB I
KONSEP DASAR BASIS DATA

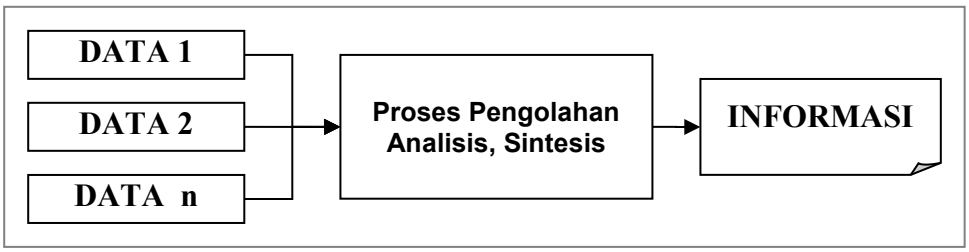
A. Pengertian Data dan Informasi

Data adalah sekumpulan fakta mengenai objek tertentu, orang, dan lain-lain. Data dapat berwujud angka, huruf, gambar, film, suara, dan sebagainya. Sedangkan **Informasi** adalah: hasil analisis dan sintesis dari data yang mempunyai makna atau hasil pengolahan dari data yang memiliki arti tertentu bagi yang membutuhkan.

Informasi adalah data yang telah diolah/diorganisasikan ke dalam bentuk yang sesuai dengan kebutuhan seseorang dalam organisasi/perusahaan. Fungsi Informasi adalah untuk membantu dalam mengambil keputusan serta mengurangi ketidakpastian dalam manajemen organisasi/perusahaan.

B. Transformasi Data Menjadi Informasi

Data hanyalah sekumpulan fakta. Namun fakta-fakta tersebut bisa memberikan suatu informasi penting setelah diolah oleh proses-proses pengolahan, analisis, dan sintesis. Pada sistem terkomputerisasi proses penyimpanan data, proses pengolahan, analisis, dan sintesis dan proses penampilan informasi bisa dilakukan oleh komputer. Gambar 1.1. menunjukkan diagram transformasi data menjadi informasi.



Gambar 1.1. Transformasi Data menjadi Informasi

Contoh data adalah: kumpulan nilai mahasiswa, nota-nota transaksi perusahaan, dan sebagainya. Data tersebut akan diolah menjadi informasi.

Contoh Informasi adalah: Nilai rata-rata mahasiswa untuk matakuliah tertentu (diperoleh dari penjumlahan seluruh data nilai suatu matakuliah dibagi dengan jumlah mahasiswa). Contoh yang lain adalah: Matakuliah yang kurang dikuasai mahasiswa (diperoleh dari analisis nilai mahasiswa), atau laporan pendapatan perusahaan (diperoleh dari analisa nota-nota transaksi perusahaan).

C. Pengertian Basis Data

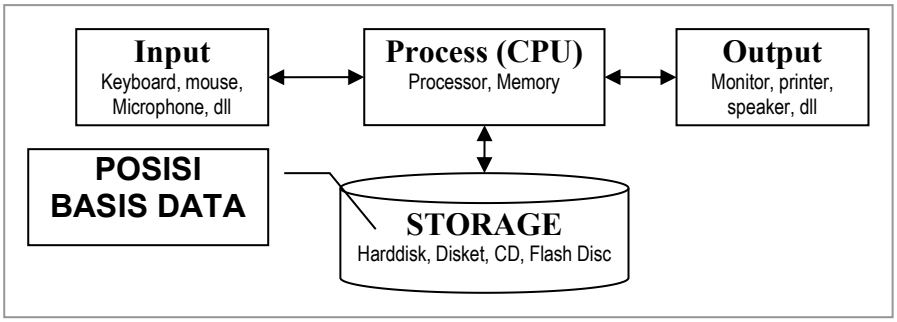
Basis Data adalah sekumpulan data terhubung yang disimpan dalam suatu media, tanpa **mengatap** satu sama lain dan tidak terjadi kerangkapan data.

Sekumpulan data tersebut disimpan dengan cara-cara tertentu sehingga mudah digunakan oleh satu atau banyak program aplikasi secara optimal **tanpa** ketergantungan dengan program *aplikas*i/software yang menggunakannya.

Data disimpan sedemikian rupa sehingga pendefinisian dan manipulasi data dapat dilakukan dengan mudah dan terkontrol (Sutanta, 1997).

D. Posisi Basis Data pada Sistem Komputer

Pada sistem komputer, basis data disimpan dalam wujud file-file atau sebuah file database. File-file database tersebut disimpan dalam media penyimpan (storage) seperti harddisk, disket, CD, flash disc, dan sebagainya.



Gambar 1.2. Posisi Basis Data pada Sistem Terkomputerisasi

E. Sistem Manajemen Basis Data (Database Management system)

Database Management System (DBMS) adalah suatu program/software komputer yang digunakan untuk membuat, memasukkan, mengubah, menghapus, memanipulasi data dan memprosesnya menjadi informasi dengan praktis dan efisien. Contoh Software DBMS adalah :

1. Dbase III+ s/d Dbase 5, Clipper, Foxbase, Foxpro (pada sistem operasi DOS)
2. Visual Foxpro, Visual Dbase, Visual Basic, Delphi, ORACLE, Power Builder, MS. Access, dsb (pada sistem operasi Windows).
3. MySQL, PostGreSQL, Kylix, dsb (pada sistem operasi Linux)

Software-software diatas lebih memberikan kemudahan jika dipakai untuk manajemen basis data dibandingkan dengan software pemrograman dasar seperti : Pascal, COBOL, BASIC, atau Fortran. Software DBMS diatas selain sudah dikhususkan untuk pengolahan basis data, biasanya juga sudah disediakan sarana untuk pengamanan data (security), manajemen pengguna (user), dan fasilitas query. Kemampuan yang lain adalah umumnya software tersebut sudah siap pakai, mudah untuk diakses oleh banyak pengguna secara bersamaan (multi user) dan dilengkapi fasilitas pengecekan *integritas data*.

Komponen utama DBMS dapat dibagi menjadi empat macam yaitu :

1. Perangkat keras (hardware) :
   Perangkat keras berupa CPU, storage, CD-ROM, monitor beserta peripheralnya seperti printer, mouse dan keyboard. Komponen ini berguna untuk pemrosesan dan menyimpan basis data
2. Perangkat Lunak (software)
   Perangkat lunak berfungsi sebagai penghubung antara pengguna dengan perangkat keras. Contoh perangkat lunak adalah : sistem operasi (Windows, DOS, Linux), Software pengolah database, dan sebagainya.

3. Data

   Data adalah isi dari basis data dan merupakan sesuatu yang paling bernilai dalam basis data
4. Pengguna

   Pengguna adalah orang-orang yang memanfaatkan, mengelola basis data

F. Database Administrator (DBA)

Database Administrator (DBA) atau Administrator Basis Data merupakan salah satu jabatan dalam organisasi pengelola sistem terkomputerisasi. DBA adalah orang yang bertanggung-jawab terhadap pengelolaan basis data. Beberapa pekerjaan yang dilakukan seorang DBA adalah :

1. Membangun basis data
   a. Mendefinisikan basis data
   b. Menentukan isi basis data
   c. Menentukan sekuritas basis data
2. Memantau kinerja sistem
3. Merencanakan *back up* dan *recovery* basis data
4. Mengikuti perkembangan produk seperti mengikuti perkembangan DBMS yang baru, berikut *tools* dan perangkat pendukungnya.

--- 000 ---

BAB II
MODEL DAN ARSITEKTUR SISTEM BASIS DATA

Model basis data menyatakan hubungan antar rekaman yang tersimpan dalam basis data. Beberapa literatur menggunakan istilah **struktur data logis** untuk menyatakan keadaan ini. Model dasar yang paling umum ada 3 macam, yaitu(Kadir, 2001)

A. Hirarki
B. Jaringan/Network
C. Relasional

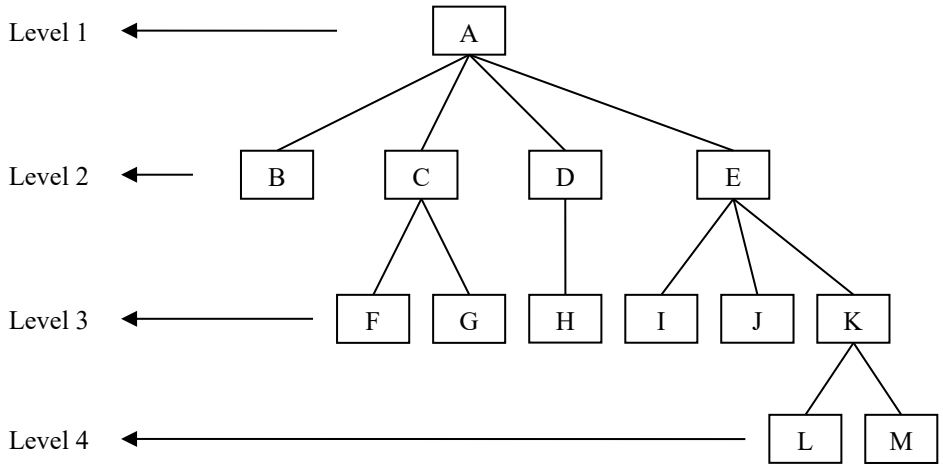
Model yang lebih baru dikembangkan oleh sejumlah periset, yang dapat disebut sebagai sistem *pascarelasional*. Beberapa di antaranya merupakan pengembangan dari pendekatan relasional, sedangkan yang lain benar-benar menggunakan pendekatan yang sama sekali berbeda. Beberapa nama yang sedang dikembangkan oleh para periset, antara lain:

1. DBMS deduktif
2. DBMS pakar
3. DBMS semantic
4. DBMS berorientasi objek
5. DBMS relasional universal

Beberapa produk sistem berorientasi objek telah beredar di pasar, antara lain OpenODB (Hawlett Packard Corporation) dan ObjectStore (Object Design Corporation). Beberapa produk di lingkungan PC juga menuju ke arah ini.

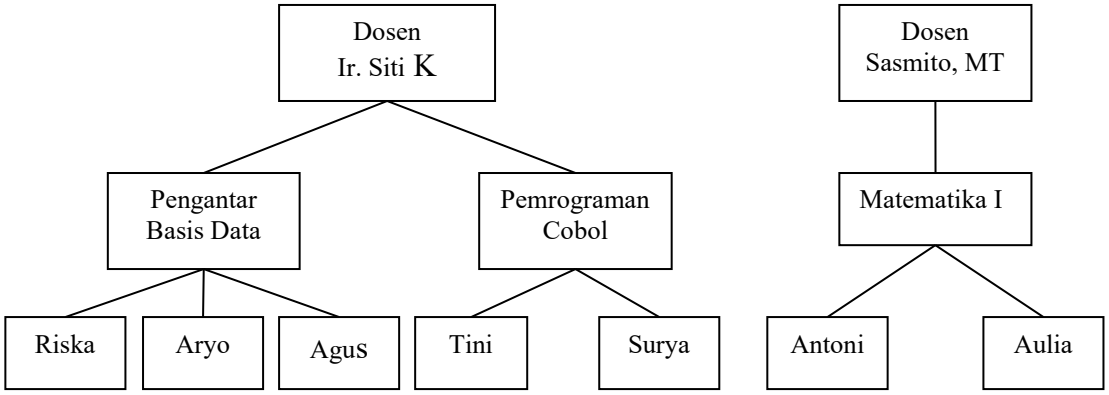
A. Model Data Hierarchycal & Sistem IMS

Model hierarchical/hirarkis biasa disebut model pohon, karena meyerupai pohon yang dibalik. Model ini menggunakan pola hubungan orangtua–anak. Setiap simpul (biasa digambarkan dengan lingkaran atau kotak) menyatakan sekumpulan medan. Simpul yang terhubung ke simpul pada level di bawahnya disebut **orangtua.** Setiap orangtua bisa memiliki satu (hubungan 1:1) atau beberapa anak (hubungan 1:m), tetapi setiap anak hanya memiliki satu orangtua. Simpul-simpul yang dibawah simpul orangtua disebut simpul **anak**. Simpul orangtua yang tidak memiliki orangtua disebut **akar.** Simpul yang tidak memiliki anak disebut **daun**. Adapun hubungan antara anak dan orangtua disebut **cabang**. Gambar 2.1. memperlihatkan contoh model hirarkis, yang terdiri atas 4 level dan 13 simpul.



Gambar 2.1. Contoh Model Hirarkis

Pada contoh diatas, A berkedudukan sebagai akar, dan berkedudukan sebagai orangtua dari simpul B, C, D dan E. Keempat simpul yang disebutkan belakangan inidisebut sebagai anak simpul A. C juga bisa berkedudukan sebagai orantua, yaitu orangtua F dan G. Adapaun simpul F, G, H, I, J , L, dan M disebut sebagai daun. Contoh yang lebih konkret dapat dilihat pada gambar 2.2. Gambar ini memperlihatkan hubungan dosen dan kelas yang diampu, serta mahasiswa yang mengikuti kelas masing- masing.



Gambar 2.2. Contoh Model Hirarkis

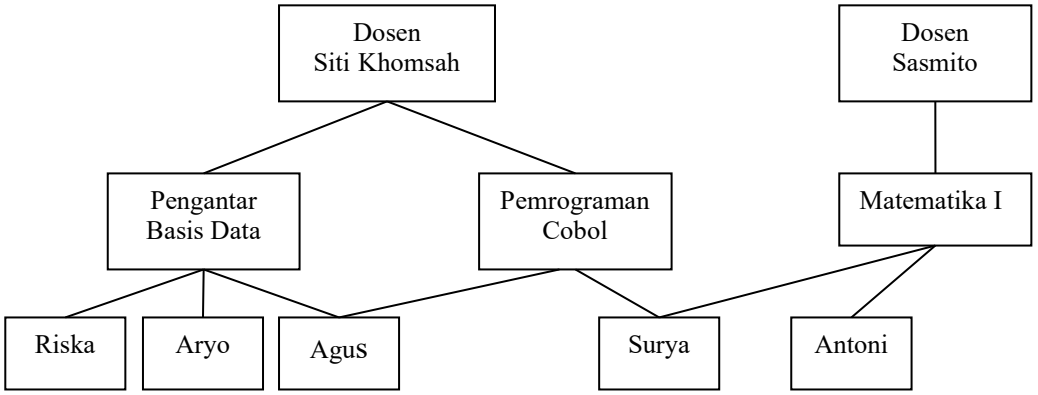
Contoh produk DBMS yang menggunakan model hirarkis adalah IMS (Information Management System) yang dikembangkan oleh dua perusahaan : IBM dan Rockwell International Corporation.

B. Model Data Network (Jaringan) & Sistem DBTG

Model data network/jaringan di standarisasi pada tahun 1971 oleh Data Base Task Group (DBTG). Itulah sebabnya disebut sistem

DBTG. Model ini juga disebut model CODASYL (Conference on Data System Languages), karena DBTG adalah bagian dari CODASYL.

Model ini mernyerupai model hirarkis, dengan perbedaan suatu simpul anak bisa memiliki lebih dari satu orangtua. Oleh karena sifatnya yang demikian, model ini bisa menyatakan hubungan 1:1 (satu orangtua punya satu anak), 1:M (satu orang tua punya banyak anak), maupun N:M (beberapa anak bisa mempunyai beberapa orangtua). Pada model jaringan, orangtua disebut **pemilik** dan anak disebut **anggota.** Gambar 2.3. merupakan model jaringan yang didasarkan oleh model hirarkis dari gambar 2.2. diatas.



Gambar 2.3. Model Data Jaringan/Network

.

C. Model Data Relational & Arsitektur Sistem R

Model relasional merupakan model yang paling sederhana sehingga mudah digunakan dan dipahami oleh pengguna, serta merupakan yang paling populer saat ini. Model ini menggunakan

sekumpulan tabel berdimensi dua, dengan masing-masing tabel tersusun atas *tupel* atau baris dan atribut. Tabel dirancang sedemikian rupa sehingga dapat menghilangkan kemubaziran data dan menggunakan kunci tamu untuk berhubungan dengan tabel lain. DBMS yang bermodelkan relasional biasa disebut RDBMS (*Relational Database Management System*).

Tabel dibawah ini memperlihatkan istilah relasi, baris, dan atribut dan padanannya dengan istilah-istilah lain yang populer di kalangan pemrogram dan sejumlah pengguna (terutama yang bekerja dengan SQL).

Tabel 2.1. Padanan Istilah Relasi, *Tupel* dan Atribut.

Model Relasional	**Pemrogram**	**Pengguna**
Relasi	Berkas	Tabel
Tupel (baris)	Rekaman	Baris
Atribut	Medan	Kolom

Catatan:

- Meskipun istilah “relasi” memiliki arti yang sama dengan “tabel”, tetapi istilah “relasi” pada sejumlah literatur sering disebut saat membicarakan *struktur logis*, dan tabel adalah nama yang sering disebut pada basis data fisi.
- Pada buku ini, istilah “tabel” terkadang digunakan untuk menggantikan “relasi”, atau sebaliknya.

Tabel dibawah ini merupakan bentuk relasional berdasarkan contoh model hirarkis dan jaringan di depan.

Tabel 2.2. Model Relasional dari Model Hirarkis Pada Gambar 2.3.

Nama Dosen	Kelas	Mahasiswa
Ir. Siti khomsah	Pengantar Basis Data	Agus
Ir. Siti khomsah	Pengantar Basis Data	Aryo
Ir. Siti khomsah	Pengantar Basis Data	Riska
Ir. Siti khomsah	Pemrograman Cobol	Surya
Ir. Siti khomsah	Pemrograman Cobol	Agus
Sasmito, MT	Matematika I	Antoni
Sasmito, MT	Matematika I	Surya

Pada prakteknya, tabel pada gambar diatas akan dinormalisasikan sehingga akan berbentuk beberapa tabel yang saling berhubungan. Contoh model relasional seperti ini dapat dilihat pada gambar 2.4. Pada gambar ini terdapat tiga buah tabel. Tabel yang terbawah menggunakan kunci tamu berupa nomor mahasiswa (NO_MHS) dan kode matakuliah (KODE_MK) untuk menghubungkan diri ke kedua tabel diatasnya. Dengan kata lain, berdasarkan data pada tabel terbawah, informasi seperti nama mahasiswa (NAMA_MHS) dan nama matakuliah (NAMA_MK) bisa diperoleh.

Ada beberapa sifat yang melekat pada suatu tabel:

1. Tidak ada baris (record) yang kembar
2. Urutan baris (record) tidak penting karena baris-baris dapat dipandang dalam sembarang urutan.
3. Setiap atribut memiliki nama yang unik.
4. Letak atribut bebas (urutan atribut tidak penting).

5. Setiap atribut memiliki nilai tunggal dan jenisnya sama untuk semua baris.

NO_MHS	NAMA_MHS
55	Ahmad
56	Lusi
57	Bono

KODE_MK	NAMA_MK
DB001	Pengantar Basis Data
DB002	Basis Data Lanjut
PI001	Teknik Multimedia

NO_MHS	KODE_MK	NILAI
55	DB001	A
55	PI001	B
56	DB001	B
57	DB001	A
57	DB002	A

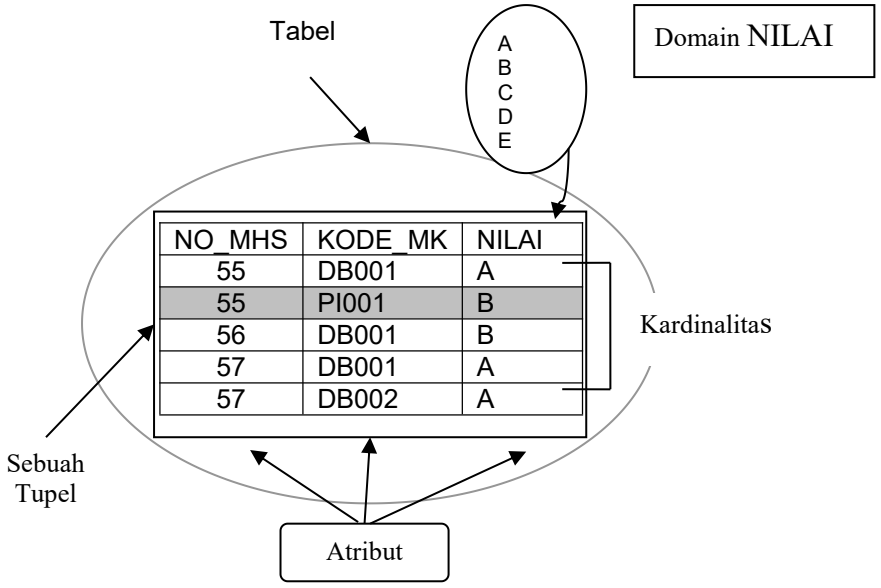
Gambar 2.4. Beberapa Tabel pada Model Relasional.

Pada model relasional, jumlah baris (record) suatu tabel disebut ***kardinalitas*** dan jumlah atribut suatu tabel disebut ***derajat*** (degree) atau terkadang disebut ***arity***. Tabel yang berderajat satu (hanya memiliki satu atribut) disebut ***unary***. Tabel yang berderajat dua disebut ***binary*** dan tabel yang berderajat tiga disebut ***ternary***. Tabel yang berderajat n disebut ***n-ary.***

Istilah lainnya yang terdapat pada model relasional adalah **domain**. Domain adalah himpunan nilai yang berlaku bagi suatu atribut.

Sebagaimana dicantumkan diatas, baris-baris (record) yang terdapat pada suatu tabel tidak ada yang kembar. Sesungguhnya bagian yang menyebabkan tidak adanya tupel yang kembar adalah yang disebut kunci primer. Sebagi contoh, pada tabel yang terlihat

pada gambar 2.5. di bawah ini, kunci primer adalah gabungan dari kunci atribut NO_MHS dan KODE_MK. Kunci gabungan semacam ini disebut kunci komposit.



Gambar 2.5. Tabel dan Komponen-Komponen Penyusunnya

--- 000 ---

BAB III
TABEL DALAM BASIS DATA

A. Pendahuluan

Basis Data pada sistem terkomputerisasi sudah “distrukturkan” menjadi tabel-tabel data. Tabel-tabel berfungsi sebagai tempat penyimpan data dan untuk mengorganisir data.

Pada model database zaman dahulu (tahun 1980-an) sebuah tabel sudah disebut sebagai sebuah database. Pengertian tersebut saat ini sudah tidak dipakai. Saat ini sebuah database berisi beberapa buah tabel yang saling berhubungan. Contoh: Database AKADEMIK berisi Tabel Mahasiswa, Tabel DosenWali, Tabel Matakuliah, Tabel KRS, dan seterusnya.

Pendefinisian tabel-tabel dalam sebuah database tergantung dari sistem. Satu sistem akademik di sebuah perguruan tinggi berbeda dengan di perguruan tinggi yang lain. Sangat mungkin komposisi tabel-tabel dalam database masing-masing perguruan tinggi berbeda-beda.

B. Konsep Dasar Tentang Tabel

1. Pengertian Tabel

Tabel adalah blok data dasar yang paling umum digunakan dalam sistem basis data. Tabel juga diistilahkan dengan nama “relasi” (sudah dibahas pada bab sebelumnya).

2. Fungsi Tabel

Fungsi utama tabel adalah untuk menyimpan data dan manajemen data. Fungsi yang lain adalah untuk mengelompokkan data.

3. Wujud Tabel

Wujud tabel sama seperti tabel-tabel yang sering kita lihat. Contoh :

Tabel 3.1. Mahasiswa

Nomhs	Nama	Alamat	Tgl-lahir	Kode dosen wali	IPK
020001	Adhi Patrio	Jl. Paris 34 Yogyakarta	10/01/79	D122	3,22
020002	Widi Sanjaya	Jl. Kemuning 32 Yogyakarta	6/10/81	D112	2,67
020003	Anjar Permana	Jl. Beo 44 Yogyakarta	13/4/80	D112	3,45

Tabel 3.2. Dosenwali

Kode dosen wali	Nama	Alamat	Telepon
D110	Ir. Wisnu Aji, MBA	Jl. Mangga 4 Yogyakarta	-
D111	Drs. Baskoro, S.U	Jl. Pelem 2 Yogyakarta	0251677443
D112	Anjar Permana, M. Kom	Jl. Lumba-lumba 4 Yogyakarta	08122713311
D122	Ir. Ibrahim, MT	Jl. Solo 22 Yogyakarta	0274344566

4. **Komponen Tabel**

   a) **Field (Kolom) dan Record (Baris/tupel)**

      Field adalah kolom-kolom yang terdapat dalam sebuah tabel. Sedangkan record adalah baris-baris dalam sebuah tabel.

   b) **Field Kunci** : adalah field dalam tabel yang umumnya dipakai sebagai penanda record.

   c) **Atribut** dan **Nilai Atribut.**

      Atribut adalah judul dari sebuah field, sedangkan **nilai atribut** adalah data yang terdapat pada perpotongan baris dan kolom.

   d) **Kardinalitas**: jumlah record dalam suatu tabel

   e) **Derajat/degree**: jumlah dari kolom pada suatu tabel

      Contoh pada tabel berikut ini:

Tabel 3.2. Dosenwali

Kodedosenwali	Nama	Alamat	Telepon
D110	Ir. Wisnu Aji, MBA	Jl. Mangga 4 Yogyakarta	-
D111	Drs. Baskoro, S.U	Jl. Pelem 2 Yogyakarta	0251677443
D112	Anjar Permana, M. Kom	Jl. Lumba-lumba 4 Yogyakarta	08122713311
D122	Ir. Ibrahim, MT	Jl. Solo 22 Yogyakarta	0274344566

Penjelasan:

Field (kolom) pada tabel diatas adalah : kolom kodedosenwali, kolom nama, kolom alamat, dan kolom telepon. Sedangkan record

pada tabel itu adalah record D110, D111, D112, dan D22. Data pada field kodedosenwali bisa dipakai untuk mewakili record karena bersifat unik, tidak ada data yang ganda.

Atribut untuk kolom kodedosenwali adalah kodedosenwali. Nilai atribut dari kodedosenwali adalah: D110, D111, D112, dan D122. Atribut untuk kolom telepon adalah telepon. Sedangkan nilai atribut dari telepon adalah : – (null=kosong), 0274677433, 08122713311, dan 0274344566. Kardinalitas tabel diatas adalah 4, dan derajat/degree tabel diatas juga 4.

5. Tipe Data Dalam Tabel

Tipe data dalam tabel adalah jenis data pada tiap perpotongan baris dan kolom. Tipe data dalam tabel dibedakan antara satu kolom dengan kolom yang lain. Setiap jenis data mempunyai tipe data yang berbeda-beda. Jenis tipe data tersebut juga bervariasi tergantung DBMS pembuatnya. Berikut ini adalah tabel tipe data yang disediakan oleh 3 macam DBMS yaitu DbaseIII+, Paradox, dan Microsoft Access. Tampak bahwa variasi tipe data yang diberikan Paradox dan Microsoft Access lebih beragam daripada Dbase III+.

Tabel 3.3. Daftar Tipe Data

Tipe Data Pada Dbase III+.	Tipe Data Pada Paradox	Tipe Data Pada Microsoft Access
CHARACTER	ALPHANUMERIC	Text
NUMERIC	NUMBER	Number
	MONEY	Currency
	BYTES	Byte

	SHORT	Integer
	LONGINT	Long Integer
	BCD	Single Double Replication ID Decimal
DATE	DATE	Long Date Medium Date Short Date
	TIME	Long Time Medium Time Short Time
	TIMESTAMP	General Date
MEMO	MEMO	Memo
LOGICAL	LOGICAL	Yes/No True/False On/Off
	AUTOINCREMENT	Autonumber
	BINARY	Ole Object
	FORMATTED_MEMO	
	GRAPHIC	
	OLE	

6. Struktur Tabel

Struktur tabel adalah dokumentasi tentang suatu tabel yaitu : **Nama tabel, nama-nama field yang ada pada tabel, tipe data setiap field dalam tabel beserta ukurannya, serta penjelasan singkat tentang data yang disimpan**. Contoh berikut ini adalah struktur tabel untuk Tabel Mahasiswa dan Tabel Dosenwali

Tabel 3.4. Mahasiswa

No	Nama Field	Tipe Data	Ukuran	Kunci	Keterangan
1	Nomhs	Text	6	PK	Nomor mahasiswa
2	Nama	Text	30	-	Nama mahasiswa
3	Alamat	Memo		-	Alamat mahasiswa
4	Tgl-Lahir	Short		-	Tanggal lahir

		Date			mahasiswa
5	Kodedosenwali	Text	4	SK	Kode dosen wali per mahasiswa
6.	IPK	Decimal		-	Indek Prestasi Kumulatif setiap mahasiswa

Tabel 3.5. Dosenwali

No.	Nama Field	Tipe Data	Ukuran	Kunci	Keterangan
1	Kodedosenwali	Text	6	PK	Nomor kode dosenwali
2	Nama	Text	30	-	Nama dosenwali
3	Alamat	Memo		-	Alamat dosenwali
4	Telepon	Text	20	-	Nomor telepon dosenwali

7. Kerangka Tabel

Kerangka tabel adalah penyebutan struktur tabel secara singkat dan padat. Contoh kerangka tabel untuk dua tabel diatas adalah sbb :

Mahasiswa(Nomhs, Nama, Alamat, Tgl-Lahir, Kodedosenwali, IPK)
DosenWali(Kodedosenwali, Nama, Alamat, Telepon)

Atau bisa ditulis lebih lengkap sebagai berikut :

Mahasiswa(Nomhs text(6), Nama text(30), Alamat memo, Tgl-Lahir shortdate, Kodedosenwali text(4), IPK decimal)

DosenWali(Kodedosenwali text(4), Nama text(30), Alamat memo, Telepon text(20))

C. Aturan Penyusunan Sebuah Tabel

Penggunaan tabel bertujuan untuk menyederhanakan logika pandangan terhadap data. Oleh karena itu dibuat ketentuan dalam penyusunan sebuah tabel, yaitu :

1. Urutan baris tidak diperhatikan sehingga pertukaran baris diperbolehkan karena tidak mempengaruhi isi data pada tabel.
2. Urutan kolom tidak diperhatikan (boleh dipandang dari kolom mana saja). Identifikasi kolom dibedakan dengan nama atribut.
3. Tiap perpotongan baris dan kolom hanya berisi nilai atribut tunggal, sehingga nilai atribut ganda tidak diperbolehkan
4. Tiap baris dalam tabel harus dibedakan satu sama lain sehingga tidak mungkin ada dua baris dalam sebuah tabel memiliki nilai atribut yang sama secara keseluruhan.

D. Manipulasi Data Pada Tabel

Proses manipulasi data pada tabel dilakukan untuk mengolah data tersebut. Proses ini berfungsi untuk menghasilkan informasi yang berguna berdasar data yang ada. Proses pengolahan data ada 4 macam yaitu :

1. Menampilkan seluruh data atau data tertentu saja dari dalam tabel (Select)
2. Memperbaharui data yang tersimpan dalam tabel (Update)
3. Mengisi/Menambah data yang telah tersimpan dalam tabel (insert)
4. Menghapus data yang telah tersimpan dalam tabel (Delete)

--- 000 ---

BAB IV
TEKNIK NORMALISASI

A. Pendahuluan

Kegunaan utama sistem basis data adalah agar pemakai mampu menyusun suatu pandangan abstraksi dari data. Gambaran tentang data tidak lagi memperhatikan kondisi sesungguhnya bagaimana suatu data disimpan dalam harddisk, tetapi pada bagaimana data tersebut digambarkan menyerupai kondisi yang dihadapi pemakai sehari-hari. Pemakai database dapat dikelompokkan menjadi tiga tingkatan dalam menggambarkan suatu database/basis data yaitu :

- Level Phisik
- Level Konseptual
- Level Pandang Pemakai

1. Level Phisik

Level ini merupakan level abstraksi paling rendah menggambarkan bagaimana data disimpan dalam kondisi sebenarnya. Level ini tentu paling kompleks, dan dibahas pada mata kuliah Sistem Berkas.

2. Level Konseptual

Level ini merupakan level yang lebiih tinggi yang menggambarkan data apa yang disimpan dalam database dan hubungan relasional yang terjadi antara tabel data. Level ini menggambarkan keseluruhan database. Pemakai tidak

mempedulikan kerumitan dalam struktur level phisik lagi, penggambaran cukup dengan memakai kotak, garis dan keterangan secukupnya. Level konseptual ini digunakan oleh database administrator yang memutuskan data apa yang akan dipelihara dalam satu database.

3. Level Pandang Pemakai

Level abstraksi tertinggi yang menggambarkan hanya satu bagian dari keseluruhan database. Bila pada level konseptual data merupakan suatu kumpulan besar dan kompleks, pada level ini hanya sebagian saja yang dilihat dan dipakai. Hal ini disebabkan beberapa pemakai database tidak membutuhkan semua isi database. Level ini sangat dekat dengan pengguna. Setiap pengguna butuh sebagian dari database. Ada beberapa kelompok user dengan pandangan berbeda butuh data dalam database. Misalnya pengguna di bagian **pemasaran** suatu perusahaan hanya butuh data tentang barang, transaksi penjualan dan pelanggan. Tidak butuh data tentang pemasok, gudang atau karyawan

B. Merancang Model Konseptual Database

Tugas **Database Administrator** adalah merancang model konseptual database. Model ini tidak tergantung pada aplikasi individual, tidak tergantung pada DBMS yang dipakai, tidak tergantung pada hardware yang dipakai, juga tidak tergantung pada phisikal model. Semuanya masih bersifat perancangan, belum pada penerapan.

Pada perancangan model konseptual penekanan tinjauan dilakukan pada struktur data dan relasi antar tabel. Tidak perlu dipikirkan tentang terapan dan operasi yang akan dilakukan pada database.

Pendekatan yang dilakukan pada perancangan model konseptual adalah menggunakan model data relational. Terdapat dua buah teknik yaitu :

- Teknik Normalisasi
- Teknik Entity Relationship (dibahas pada bab selanjutnya)

C. Teknik Normalisasi

Proses normalisasi merupakan proses pengelompokan data elemen menjadi tabel-tabel beserta relasinya. Hasil proses normalisasi harus diuji pada beberapa kondisi apakah ada kesulitan/*anomali* pada saat menambah, menghapus, mengubah, membaca pada satu database atau tabel. Bila ada kesulitan pada pengujian tersebut maka relasi tersebut perlu dipecah pada beberapa tabel lagi atau dengan kata lain perancangan belum mendapatkan database yang optimal.

Sebelum membahas dan mendalami tentang normalisasi lebih lanjut, ada beberapa konsep yang harus diketahui terlebih dahulu yaitu konsep tentang :

- Field/atribut kunci
- Kebergantungan fungsi (*functional dependency*)

1. Field/Atribut Kunci

Setiap tabel selalu terdapat kunci dari tabel berupa sebuah field atau satu set field yang dapat mewakili record. Misalnya nomor pegawai merupakan kunci dari tabel pegawai suatu perusahaan, setiap pencarian cukup dengan menyebutkan nomor pegawai tersebut, maka dapat diketahui nama, alamat, dan atribut lainnya mengenai seorang pegawai tersebut.

a. Candidate Key (Kunci Kandidat/Kunci Calon)

Kunci kandidat adalah satu atribut atau sekelompok atribut yang mengidentifikasikan secara unik suatu record secara spesifik pada suatu tabel. Jika satu kunci kandidat berisi lebih dari satu atribut, maka biasanya disebut sebagai kunci komposit (kunci gabungan).

Contoh: perhatikan kerangka **tabel pegawai** di bawah ini

Pegawai(No_induk, No_KTP, Nama, Tempat_Lahir, Tanggal_Lahir, Alamat, Kota)

Kunci kandidat pada tabel diatas adalah :

- No_Induk : karena unik, tidak mungkin ganda
- No_KTP: Karena unik juga dan tidak mungkin ganda. Tetapi ada kemungkinan

  ada pegawai yang tidak punya KTP

- Nama : bisa dipakai sebagai kunci, kecuali ada 2 orang atau lebih yang mempunyai nama sama disimpan dalam tabel.
- Nama + tanggal_lahir : bisa dipakai sebagai kunci dan lebih kecil kemungkinannya ada orang yang bernama sama dan bertanggal lahir sama pula.
- Alamat dan Kota : tidak bisa dijadikan kunci karena sering terjadi ketidak-unikan

b. Primary Key (Kunci Primer)

Primary key adalah suatu atribut atau satu set minimal atribut yang tidak hanya mengidentifikasi secara unik suatu kejadian spesifik tapi juga dapat mewakili setiap kejadian dari suatu *entity*. Setiap kunci kandidat punya peluang mejadi primary key tetapi sebaiknya dipilih satu saja yang dapat mewakili secara menyeluruh terhadap entity yang ada. Contoh :

- No_induk : bersifat unik tidak mungkin ganda dan mewakili secara menyeluruh terhadap entity pegawai, dan setiap pegawai selalu mempunyai nomor induk.
- No_KTP : bersifat unik dan tidak mungkin ganda, tetapi kurang baik untuk kunci primer karena ada kemungkinan pegawai tidak memiliki KTP atau memiliki KTP ganda. Serta data No_KTP hanya bersifat data pelengkap dan tidak berhubungan langsung dengan identitas kepegawaian dalam perusahaan.

c. **Alternate Key (Kunci Alternatif)**

Alternate Key adalah kunci kandidat yang tidak dipakai sebagai primary key. Sering kali kunci alternatif dipakai sebagai kunci pengurutan dalam laporan.

d. **Foreign Key (Kunci Tamu)**

Foreign Key adalah satu atribut (atau satu set atribut) yang melengkapi satu relationship yang menunjukkan induknya. Kunci tamu ditempatkan pada entity anak dan atributnya harus sama dengan kunci primary induk yang direlasikan dengannya. Hubungan antara entity induk dengan anak adalah hubungan satu lawan banyak (one to many relationship).

2. Kebergantungan Fungsi

Definisi dari Kebergantungan Fungsi (functional dependency) adalah :

> Jika ada tabel bernama **Tabel T**, maka atribut **Y pada Tabel T** akan bergantung **fungsi pada atribut X pada Tabel T jika dan hanya jika : setiap nilai atribut X dalam Tabel T** mempunyai hubungan dengan tepat satu nilai atribut **Y** didalam **Tabel T** (dalam setiap satu waktu).

Contoh kerangka **tabel pegawai** berisi atribut :

Pegawai(No_induk, No_KTP, Nama, Tempat_Lahir, Tanggal_Lahir, Alamat, Kota)

Isi dari atribut *Nama* bergantung pada *No_induk*. Jadi dapat dikatakan bahwa atribut *Nama* bergantung secara fungsi pada *No_Induk* dan *No_Induk* menunjukkan secara fungsi *Nama*. Jika

anda mengetahui *No_Induk* pegawai, maka anda dapat menentukan *Nama* pegawai tersebut beserta alamatnya serta atribut lainnya. Notasi untuk kebergantungan fungsi ini adalah :

No_Induk → Nama
atau,
Nama = f(No_Induk)

Hal yang sama juga berlaku untuk atribut lainnya.

D. Proses Normalisasi

Proses normalisasi merupakan proses bertahap dimulai dari tahap 0 hingga tahap 3 sebagai berikut :

Langkah 0. Mendapatkan Data di Lapangan.

Data di lapangan berupa faktur-faktur, catatan, naskah-naskah, dan sebagainya. Bentuk data ini merupakan kumpulan data yang akan direkam dengan tidak ada keharusan mengikuti format aturan tertentu. Dapat saja data tidak lengkap atau terduplikasi. Data dikumpulkan apa adanya sesuai dengan kedatangannya. Contoh kongkret adalah pada pengamatan sebuah **dokumen dasar** yang diperoleh dari hasil survey di lapangan. Contoh dokumen dasar : **Faktur Penjualan Barang**. Berikut contoh faktur-faktur penjualan barang dari lapangan :

FAKTUR PENJUALAN BARANG
PT. JANATECHNIC
Jl. Pingit XX
Yogyakarta

Kode Pelanggan : P002 **Tanggal** : 06/10/02
Nama Pelanggan : Ir. Anjani Adi **Nomor Faktur** : F102

Kode Barang	Nama Barang	Jumlah	Harga	Total
B001	Kulkas JVC	3	1.500.000	4.500.000
B002	TV Sony	5	1.000.000	5.000.000
			Total Faktur :	**9.500.000**

Tanggal Pelunasan : 20/10/02

FAKTUR PENJUALAN BARANG
PT. JANATECHNIC
Jl. Pingit XX
Yogyakarta

Kode Pelanggan : P004 **Tanggal** : 08/10/02
Nama Pelanggan : Ir. Adi Sumanto **Nomor Faktur** : F106

Kode Barang	Nama Barang	Jumlah	Harga	Total
B001	Kulkas JVC	4	1.500.000	6.000.000
B004	TV Sharp	5	800.000	4.000.000
		Total Faktur	**:**	**10.000.000**

Tanggal Pelunasan : 20/10/02

Gambar 5.1 Contoh Faktur Penjualan Barang

Langkah 1. Membuat Bentuk Tidak Ternormalisasi

Bentuklah faktur di atas menjadi bentuk tidak ternormalisasi dengan mencantumkan semua field data yang ada :

Nomor Faktur	Kode Pelanggan	Nama Pelanggan	Kode Barang	Nama Barang	Tanggal	Pelunasan	Jumlah	Harga	Total	Total Faktur
F101	P002	Ir Anjani	B001 B002	Kulkas JVC TVSony	06/10/02	20/10/02	3 5	1.500.000 1.000.000	4.500.000 5.000.000	9.500.000
F106	P004	Ir Adi Sumanto	B001 B004	Kulkas JVC TV Sharp	08/10/02	2-/10/02	4 5	1.500.000 800.000	6.000.000 4.000.000	10.000.000

Cara membuat tabel diatas adalah dengan menuliskan semua data yang akan direkam, bagian yang double tidak perlu dituliskan. Terlihat record-record yang tidak lengkap, sulit untuk

membayangkan bagaimana bentuk record yang harus dibentuk untuk merekam data tersebut.

Langkah 2. Bentuk Normal Kesatu.

Bentuk normal kesatu mempunyai ciri bahwa setiap data dibentuk dalam flat tabel (tabel datar/rata). Data dibentuk dalam satu record demi satu record dan nilai dari field-field berupa “atomic value”. Tidak ada set atribut yang berulang-ulang atau atribut bernilai ganda (multi value). Tiap-tiap field hanya satu pengertian bukan merupakan kumpulan kata yang memiliki arti mendua, hanya satu arti saja dan juga bukanlah pecahan kata-kata sehingga artinya lain. Atomic adalah satuan terkecil yang masih memiliki sifat induknya dan bila dipecah lagi maka ia tidak lagi memiliki sifat induknya.

Bentuklah tabel tidak ternormalisasi diatas menjadi bentuk normal kesatu dengan memisah-misahkan data pada field-field yang tepat dan bernilai atomic, juga seluruh record harus lengkap adanya.

Dengan bentuk normal kesatu ini telah dapat dibuat satu tabel dengan 11 field yaitu *nomor faktur, kode pelanggan, nama pelanggan, kode barang, nama barang, tanggal, tanggal pelunasan, jumlah, harga, total*, dan *total faktur.*

Nomor Faktur	Kode Pelanggan	Nama Pelanggan	Kode Barang	Nama Barang	Tanggal	Pelunasan	Jumlah	Harga	Total	Total Faktur
F101	P002	Ir Anjani	B001	Kulkas JVC	06/10/02	20/10/02	3	1.500.000	4.500.000	9.500.000
F101	P002	Ir Anjani	B002	TVSony	06/10/02	20/10/02	5	1.000.000	5.000.000	9.500.000
F106	P004	Ir Adi Sumanto	B001	Kulkas JVC	08/10/02	2-/10/02	4	1.500.000	6.000.000	10.000.000
F106	P004	Ir Adi Sumanto	B004	TV Sharp	08/10/02	2-/10/02	5	800.000	4.000.000	10.000.000

Namun bentuk normal kesatu ini mempunyai banyak kelemahan dan kesulitan yaitu :

a. **Pada proses inserting/penyisipan**

   Kita tidak dapat memasukkan **kode** dan **nama** pelanggan saja tanpa transaksi pembelian sehingga pelanggan baru dapat masuk harus dengan transaksi pembelian.

b. **Pada proses deleting/penghapusan**

   Bila satu record di atas dihapus, misalnya nomor faktur F106, maka berakibat pula menghapus data barang B004, padahal data barang tersebut masih dibutuhkan.

c. **Pada proses updating/perbaikan/pembaharuan**

   Kode dan nama pelanggan terlihat ditulis berkali-kali, bila suatu ketika terjadi perubahan nama pelanggan, maka harus mengganti di semua record yang mengandung data tersebut. Bila ada yang terlewat tidak ikut diperbaharui, maka akan terjadi kesalahan/inkonsistensi data. (Data tidak konsisten lagi).

d. **Terjadi redudancy / kerangkapan data**

   Field *total* di atas merupakan redudancy karena setiap kali *harga* dikalikan dengan *jumlah* akan menghasilkan *total*. Maka field tersebut harus dibuang, bila tidak dibuang akan mengakibatkan inkonsistensi data. Ketidakkonsistenan data disebabkan jika ada perubahan *harga* maka hanya data *harga* saja yang diubah, data *total* tidak maka nilai *total* tidak sama dengan *jumlah* x *harga.*

Langkah 3. Bentuk Normal Kedua

Bentuk normal kedua memiliki syarat yaitu bentuk data telah memenuhi kriteria bentuk normal kesatu. Atribut bukan kunci haruslah bergantung secara fungsi pada kunci utama/primary key. Sehingga untuk membentuk normal kedua haruslah sudah ditentukan kunci-kunci field. Kunci field haruslah unik dan dapat mewakili atribut lainnya yang menjadi anggotanya.

Pembentukan normal kedua dengan mencari kunci-kunci field yang dapat dipakai sebagai patokan dalam pencarian dan bersifat unik. Kunci kandidat yang bisa diambil yaitu: **nomor faktur, kode pelanggan, kode barang**

Bentuklah tiga tabel dengan kunci tersebut dan lihatlah kebergantungan fungsional field-field lain terhadap field kunci, maka didapatkan tabel sebagai berikut :

Tabel Barang

Kode Barang(*)	Nama Barang
B001	Kulkas JVC
B002	TVSony
B004	TV Sharp

Tabel Pelanggan

Kode Pelanggan(*)	Nama Pelanggan
P002	Ir Anjani
P004	Ir Adi Sumanto

Tabel Faktur

Nomor Faktur(*)	Kode Pelanggan(**)	Kode Barang(**)	Tanggal	Pelunasan	Jumlah	Harga	Total Faktur
F101	P002	B001	06/10/02	20/10/02	3	1.500.000	9.500.000
F101	P002	B002	06/10/02	20/10/02	5	1.000.000	9.500.000
F106	P004	B001	08/10/02	2-/10/02	4	1.500.000	10.000.000
F106	P004	B004	08/10/02	2-/10/02	5	800.000	10.000.000

Keterangan :

* Kunci Primer dari tabel

** Kunci Penghubung/tamu dari tabel terhadap kunci primer di tabel induknya.

Dengan pemecahan seperti di atas maka sebagian dari pertanyaan pengujian pada bentuk normal kesatu yaitu *inserting, deleting, updating* telah terjawab. *Kode* dan *nama* pelanggan dapat masuk kapanpun tanpa harus ada transaksi pada tabel faktur, cukuplah dibuka Tabel Pelanggan dan sisipkan satu record baru. Demikian juga pada saat update dan delete pada Tabel Pelanggan dan Barang.

Namun permasalahan masih ada yaitu pada tabel Faktur :

- Field Jumlah dan Harga pada tabel tersebut tidak bergantung penuh pada kunci primer nomor faktur. Ia bengantung fungsi pada kode barang. Hal ini disebut sebagai kebergantungan transitif dan haruslah dipisahkan menjadi dua tabel.
- Masih ada redudancy, yaitu misalnya ada penambahan data nota yang terdiri dari 4 macam barang yang dibeli, maka 4 kali juga dituliskan nomor faktur, tanggal, tanggal pelunasan, dan total. Ini harus dipisahkan bila terjadi penggandaan tulisan berulang-ulang.

Langkah 4. Bentuk Normal Ketiga

Untuk menjadi bentuk normal ketiga maka relasi haruslah dalam bentuk normal kedua dan semua atribut bukan primer tidak punya hubungan yang transitif. Dengan kata lain, setiap atribut bukan kunci haruslah bergantung hanya pada primary key secara menyeluruh.

Bentuk normal ketiga dari relasi diatas adalah sebagai berikut :

Tabel Barang

Kode Barang(*)	Nama Barang
B001	Kulkas JVC
B002	TVSony
B004	TV Sharp

Tabel Pelanggan

Kode Pelanggan(*)	Nama Pelanggan
P002	Ir Anjani
P004	Ir Adi Sumanto

Tabel Faktur

Nomor Faktur(*)	Kode Pelanggan(**)	Tanggal	Pelunasan	Total Faktur
F101	P002	06/10/02	20/10/02	9.500.000
F106	P004	08/10/02	2-/10/02	10.000.000

Tabel ItemFaktur

Nomor Faktur(**)	Kode Barang(**)	Jumlah	Harga
F101	B001	3	1.500.000
F101	B002	5	1.000.000
F106	B001	4	1.500.000
F106	B004	5	800.000

Keterangan :

* Kunci Primer dari tabel

** Kunci Penghubung/tamu dari tabel terhadap kunci primer di tabel induknya.

--- 000 ---

BAB V
TEKNIK ENTITY RELATIONSHIP

A. Pendahuluan

Pembuatan basis data didahului dengan ***pemodelan***. Pendekatan pemodelan dapat dilakukan dengan **Teknik Normalisasi** yang dimulai dari menemukan dokumen dasar hingga tersusun bentuk-bentuk tabel-tabel normal. Cara ini disebut dengan pendekatan dari **bawah ke atas** (*bottom up approach*) dimana penyusunan basis data dimulai dari proses penormalan atribut-atribut dalam satu tabel menjadi tabel-tabel penyusun database.

Pemodelan dengan metode *bottom up* ini akan berhasil baik jika diterapkan untuk perancangan basis data yang sederhana, yaitu untuk basis data dengan atribut yang sedikit. Jika atribut basis data yang akan “dinormalkan’ mencapai ratusan bahkan ribuan akan sangat menyulitkan jika memakai Teknik Normalisasi. Para ahli akhirnya menemukan metode perancangan basis data yang lain yaitu Teknik Entity Relationship.

B. Entity Relationship Diagram (E-R Diagram)

Teknik Entity Relationship merupakan penyederhanaan prosedur pemodelan/ perancangan database. Pemodelan database tidak memperhatikan atribut, tetapi memperhatikan entity-entity yang terlibat, seperti entity mahasiswa, entity dosen, entity matakuliah, dan sebagainya, barulah kemudian ditentukan jenis

atribut masing-masing entity yang terlibat. Pemodelan dengan cara ini disebut dengan pendekatan dari **atas ke bawah** (*top down approach*).

C. Tahapan Pemodelan dengan E-R Diagram

Sebelum membuat model harus dilakukan penelitian untuk mengamati sistem terlebig dahulu. Pengamatan pada sistem harus lebih ditekankan pada pengamatan aturan bisnis (bussines rule) sehingga mengetahui entitas-entitas apa saja yang ada di sistem dan mengetahui hubungan antar entitas sistem. Secara singkat tahapan pemodelan dengan metode ini sebagai berikut :

1. Menentukan entity-entity yang akan terlibat dalam basis data.
2. Menentukan hubungan antar entity yang dilibatkan dalam basis data
3. Menentukan derajat hubungan antar entity
4. Menentukan tingkat partisipasi hubungan antar entity
5. Melengkapi setiap entity dengan atribut-atribut yang sesuai sehingga diperoleh bentuk tabel normal penuh.
6. Mewujudkan tabel-tabel relasional dari diagram entity relationship yang terbentuk.

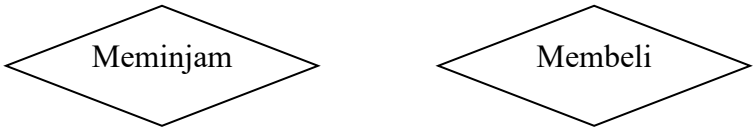
D. Lambang E-R. Diagram

Untuk menggambarkan terjadinya hubungan antar entity digunakan diagram hubungan antar entity (“entity relationship diagram”) yang biasa disingkat menjadi ER-Diagram.

Notasi yang digunakan untuk menggambarkan E-R Diagram adalah sebagai berikut :

1. Segiempat : menggambarkan entity (para subyek/obyek dalam sistem)

2. Belah ketupat : menggambarkan hubungan antar entity



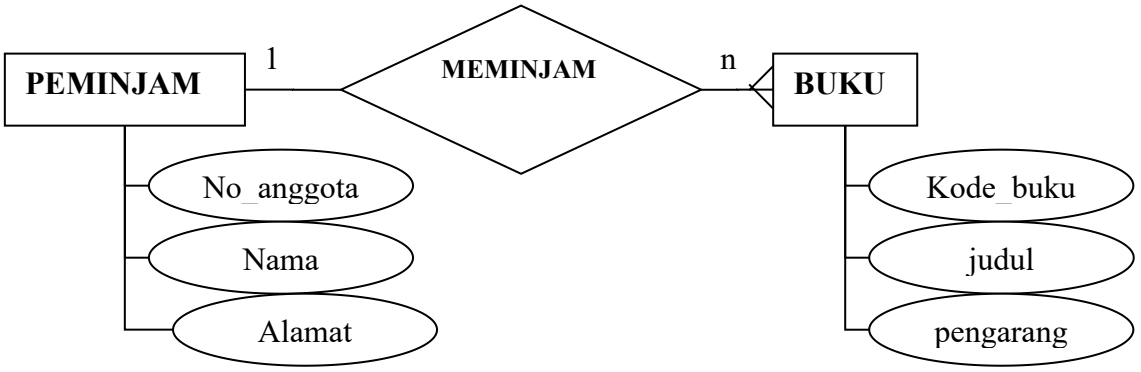
3. Elips : menggambarkan atribut bagi setiap entity

4. Garis : menggambarkan sifat hubungan/partisipasi antar entity

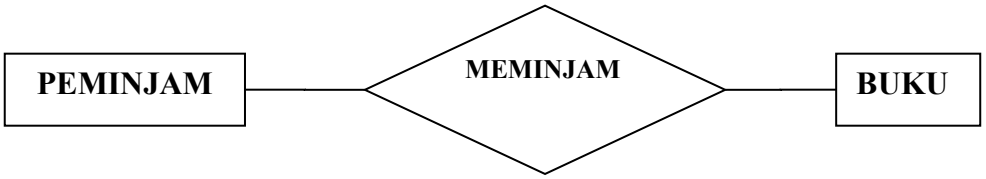
E. Contoh E-R Diagram

Gambar di bawah ini adalah contoh E-R Diagram pada basis data **PERPUSTAKAAN** yang melibatkan entity **PEMINJAM** (anggota perpustakaan) dan entity **BUKU** (buku di perpustakaan). Keduanya dihubungkan dengan hubungan **MEMINJAM**. Atribut untuk entity PEMINJAM adalah *no_anggota, nama,* dan *alamat.*

Sedangkan atribut untuk entity BUKU adalah *kode_buku, judul*, dan *pengarang*.



Dari E-R Diagram diatas jika dituliskan atribut yang banyak akan membuat diagram menjadi lebih ruwet. Penggambaran E-R Diagram bisa cukup dituliskan jenis entity dan hubungan yang terjadi saja, tanpa perlu pencantuman atribut. Sedangkan data atribut dituliskan dalam kerangka tabel entity dimana paling tidak dituliskan identitas dari tabel tersebut. Contohnya seperti pada gambar di bawah ini :



PEMINJAM(no_anggota, nama, alamat)

BUKU(kode_buku, judul, pengarang)

F. Derajat Hubungan dan Partisipasi Hubungan Antar Entity

Hubungan antar entity akan menyangkut dua komponen yang menyatakan jalinan ikatan yang terjadi yaitu : **Derajat Hubungan** dan **Partisipasi Hubungan**. **Derajat hubungan** : menyatakan jumlah anggota entity yang terlibat dalam ikatan yang terjadi. Sedangkan **partisipasi hubungan** : menyatakan sifat keterlibatan tiap anggota entity dalam ikatan terjadinya hubungan.

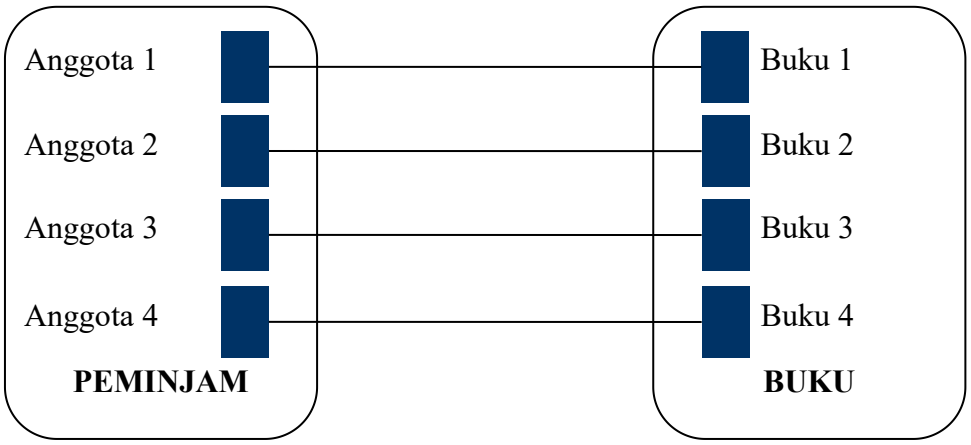
1. Derajat Hubungan

Jika entity PEMINJAM mempunyai isi : anggota1, anggota2, anggota3, ... dan entity BUKU mempunyai isi : buku1, buku2, buku3, ... Maka pasangan antara anggota entity PEMINJAM dan BUKU dapat dilakukan sesuai dengan derajat hubungannya yaitu **1:1 (satu-satu, “one to one”), 1:m (satu-banyak, “one to many”)** atau **m:n (banyak-banyak,” many to many”).**

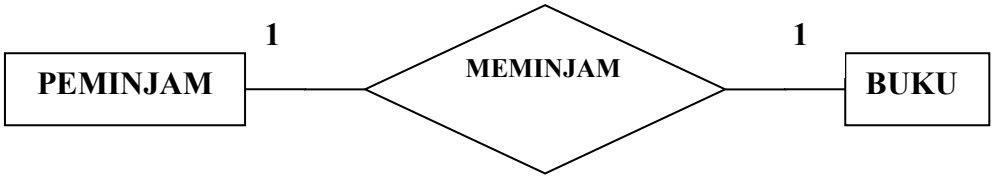
a. Derajat hubungan 1:1

Derajat hubungan antar entity 1:1 terjadi bila setiap isi dari entity PEMINJAM hanya boleh berpasangan tepat dengan satu isi dari entity BUKU. Sebaliknya tiap isi dari entity BUKU hanya boleh berpasangan dengan satu isi dari entity PEMINJAM.

Contoh : Perpustakaan yang hanya membolehkan seorang Anggota meminjam sebuah BUKU saja dan sebuah buku hanya boleh dipinjam oleh seorang anggota.



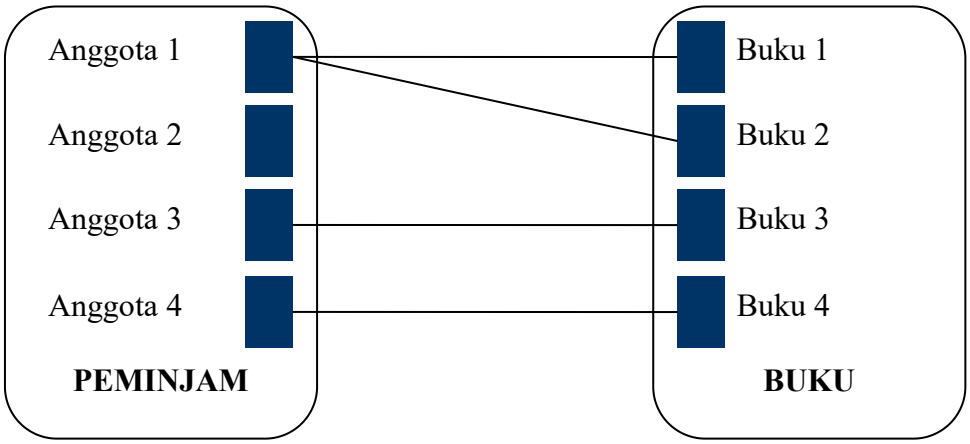
Gambar E-R diagramnya menjadi sebagai berikut :



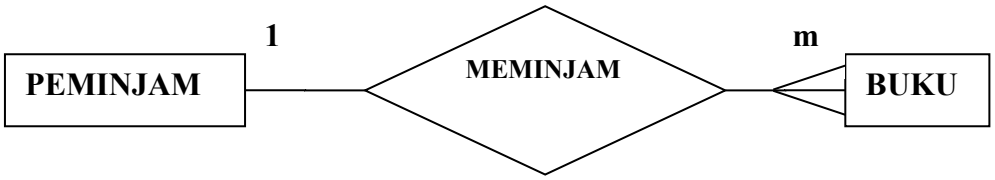
b. Derajat hubungan 1:m

Derajat hubungan antar entity 1:m terjadi bila setiap isi dari entity PEMINJAM boleh berpasangan lebih dari satu isi dari entity BUKU. Sebaliknya tiap isi dari entity BUKU hanya boleh berpasangan dengan satu isi dari entity PEMINJAM. Derajat hubungan ini mencakup juga derajat 1:1.

Contoh : Perpustakaan tersebut membolehkan seorang PEMINJAM meminjam lebih dari sebuah BUKU. Tapi sebuah BUKU hanya boleh dipinjam oleh satu PEMINJAM.



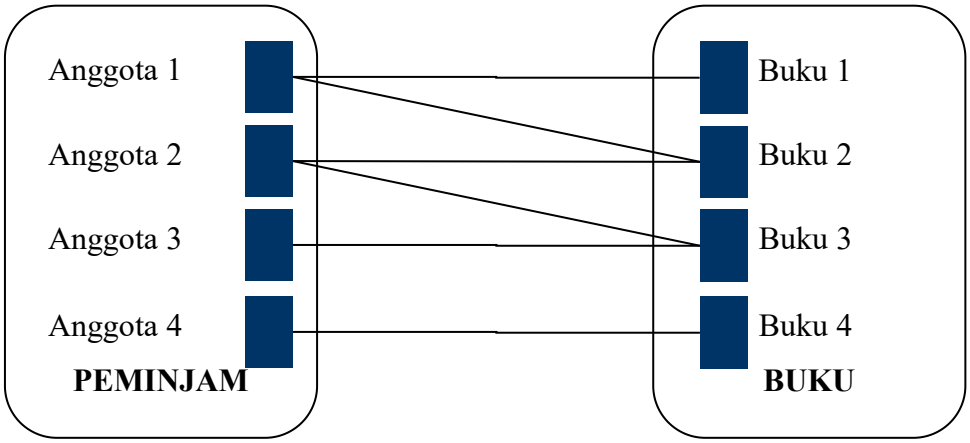
ER-Diagramnya menjadi sebagai berikut :



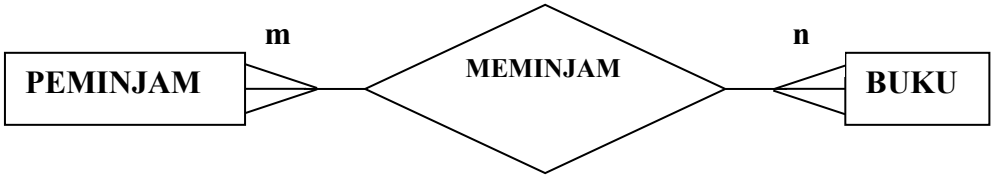
c. Derajat hubungan m:n

Derajat hubungan antar entity m:n terjadi bila setiap isi dari entity PEMINJAM boleh berpasangan lebih dari satu isi dari entity BUKU. Sebaliknya tiap isi dari entity BUKU juga boleh berpasangan dengan lebih dari satu isi dari entity PEMINJAM. Derajat hubungan ini mencakup juga derajat 1:1dan 1:m.

Contoh : Perpustakaan tersebut membolehkan seorang PEMINJAM meminjam lebih dari sebuah BUKU. Tapi sebuah BUKU juga boleh dipinjam beramai-ramai oleh lebih dari satu PEMINJAM secara bersamaan.



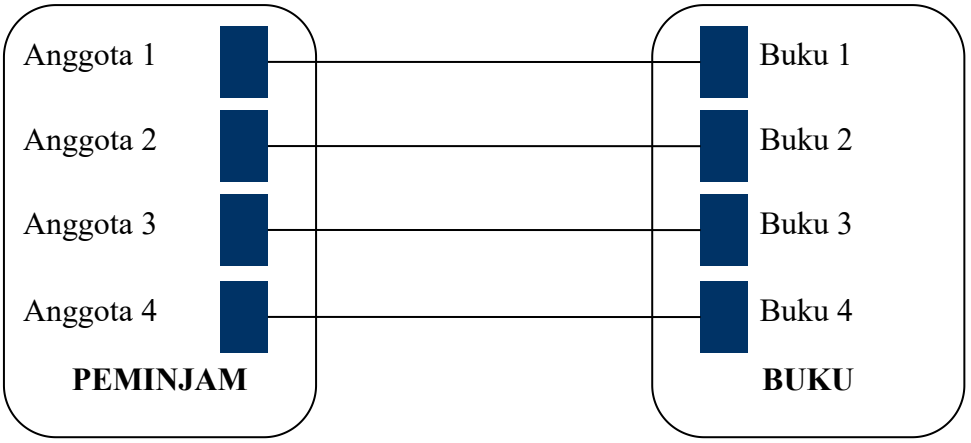
ER-Diagramnya menjadi sebagai berikut :



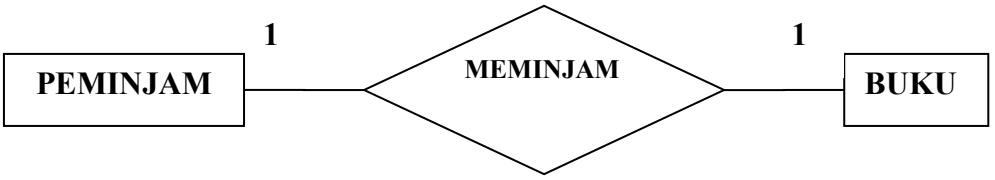
2. Partisipasi Hubungan

Partisipasi atau keterlibatan tiap isi entity dalam membentuk hubungan dapat bersifat **wajib partisipasi** atau **tidak wajib partisipasi.** Dalam pemodelan data interpretasi jenis hubungan dituliskan dalam aturan data.

Sebagai contoh hubungan antara entity PEMINJAM dengan BUKU dengan aturan baru sebagai berikut : Tiap PEMINJAM harus meminjam satu buku dan tiap BUKU harus dipinjam oleh seorang PEMINJAM maka diagram isi entitas PEMINJAM dan BUKU digambarkan sebagai berikut :

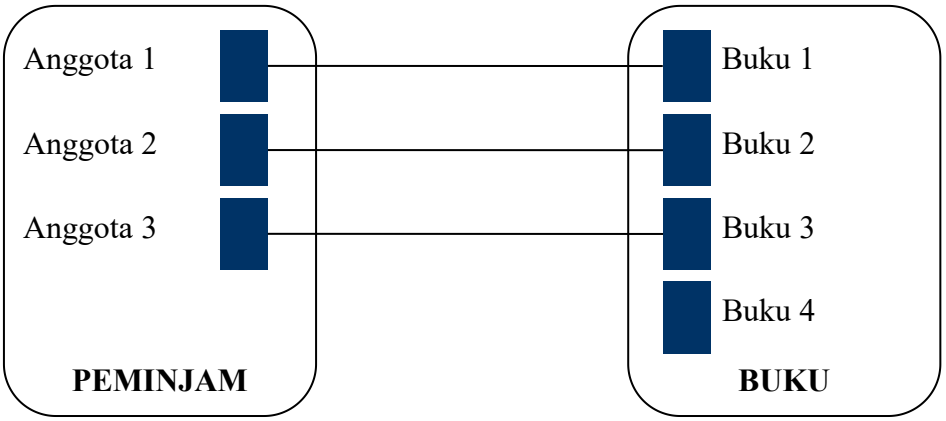


Dari gambaran diatas dapat ditarik kesimpulan bahwa derajat hubungannya adalah 1:1 sedangkan partisipasi tiap isi entity adalah wajib baik untuk isi entity PEMINJAM maupun entity BUKU. Dalam diagram E-R digambarkan sebagai berikut :

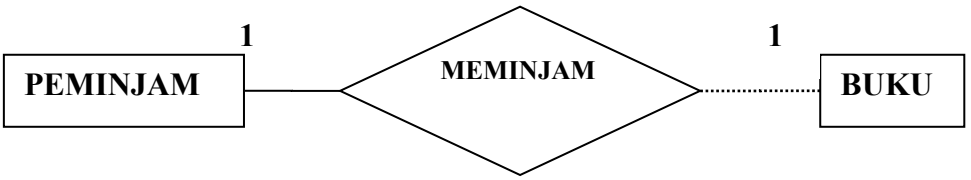


Dalam E-R diagram partisipasi wajib digambarkan dengan garis penuh pada garis hubungan antar entity. Sedangkan partisipasi tidak wajib digambarkan dengan garis putus-putus.

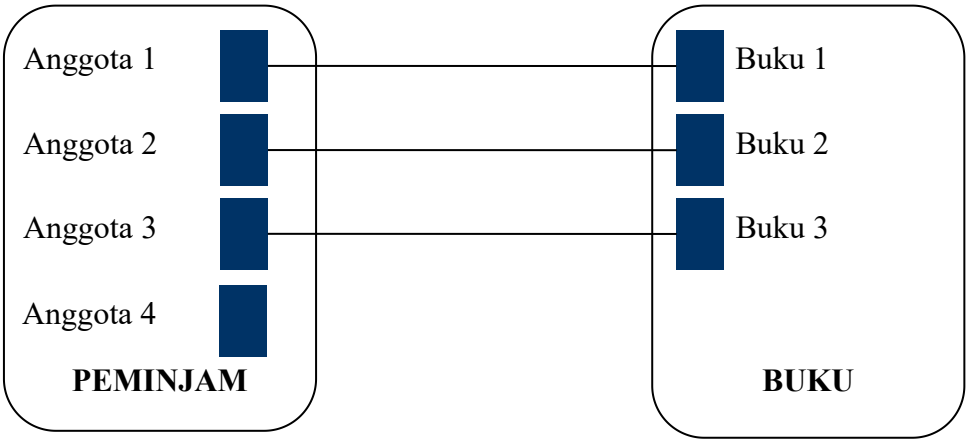
Sebagai contoh kedua, hubungan antara entity PEMINJAM dengan BUKU dengan aturan yang lebih baru lagi sebagai berikut : Tiap PEMINJAM harus meminjam satu buku dan tiap BUKU mungkin bisa dipinjam oleh seorang PEMINJAM atau mungkin juga tidak dipinjam oleh seorang PEMINJAM pun. Maka diagram isi entitas PEMINJAM dan BUKU digambarkan sebagai berikut :



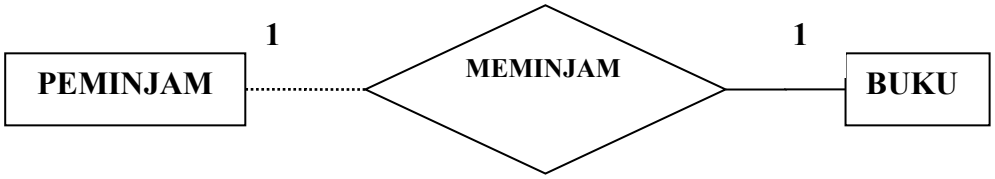
Dari gambaran diatas terlihat bahwa ada BUKU yang dipinjam (buku 1,2, dan 3) dan tidak dipinjam (buku 4), tetapi semua isi PEMINJAM wajib meminjam. Dapat ditarik kesimpulan bahwa derajat hubungannya adalah 1:1 tetapi partisipasi tiap isi kedua entity berbeda. Entity pada sisi PEMINJAM bersifat partisipasi wajib, sedangkan pada sisi BUKU tidak wajib. Dalam diagram E-R digambarkan sebagai berikut :



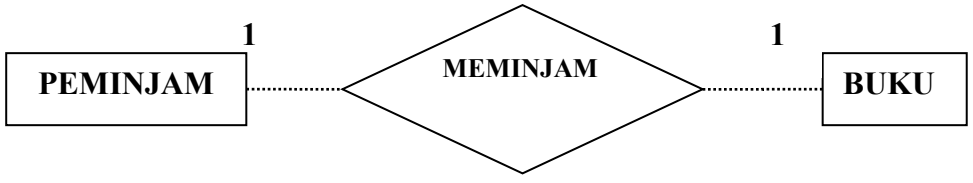
Selain dua contoh diatas mungkin sekali terjadi kasus sebaliknya jika hubungan antara entity PEMINJAM dengan BUKU diberi aturan yang lebih baru lagi sebagai berkut : Tiap PEMINJAM tidak harus meminjam buku tetapi tiap BUKU harus dipinjam oleh seorang PEMINJAM. Maka diagram isi entitas PEMINJAM dan BUKU digambarkan sebagai berikut :



Dari gambaran diatas terlihat bahwa ada PEMINJAM yang meminjam dan tidak meminjam, tetapi BUKU wajib dipinjam. Dapat ditarik kesimpulan bahwa derajat hubungannya adalah 1:1 tetapi partisipasi tiap isi kedua entity juga berbeda. Entity pada sisi BUKU bersifat partisipasi wajib, sedangkan pada sisi PEMINJAM tidak wajib. Dalam diagram E-R digambarkan sebagai berikut :



Kasus lain : Jika terdapat gambar E-R Diagram hubungan PEMINJAM dengan BUKU sebagai berikut :



dapat disimpulkan bahwa pada kedua entity : setiap PEMINJAM boleh meminjam atau tidak meminjam sebuah BUKU dan sebuah

BUKU boleh dipinjam atau tidak dipinjam oleh seorang PEMINJAM. Partisipasi kedua entitas **bersifat tidak wajib.**

Proses penentuan partisipasi hubungan antar entity juga berlaku untuk jenis hubungan yang lain baik **one to many** atau **many to many**.

BAB VI
KONVERSI E-R DIAGRAM MENJADI TABEL-TABEL RELATIONAL

Pendahuluan

Pemodelan E-R diagram bertujuan untuk menghasilkan tabel-tabel relational pada database berukuran besar. Pembentukan tabel-tabel harus juga selalu berpedoman pada ketentuan-ketentuan tentang cara penyusunan tabel. Sehingga akan diperoleh susunan tabel entity yang merupakan tabel-tabel normal penuh dan terbebas dari adanya data rangkap sehingga terhindar dari inkonsistensi data saat dilakukan operasi insert, update, dan delete.

Untuk mencapai tujuan di atas dalam penyusunan tabel-tabel entity harus memperhatikan hubungan antar entity yang terjadi, yaitu derajat dan partisipasi hubungan. Berikut ini akan dibahas penyusunan tabel entity dengan memperhatikan derajat dan partisipasi hubungan antar entity.

A. Representasi Hubungan 1:1

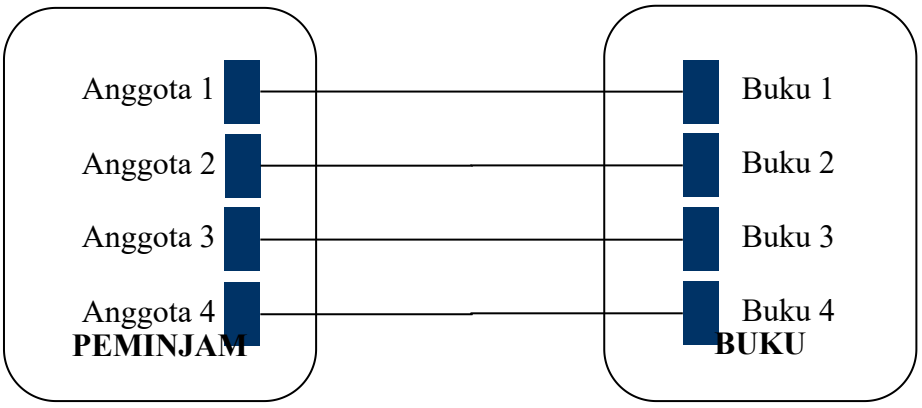
Misalnya suatu perpustakaan memberikan aturan peminjaman bagi PEMINJAM terhadap BUKU. Aturan yang berlaku sesuai representasi hubungan 1:1 adalah : **satu PEMINJAM perpustakaan meminjam satu BUKU, dan satu buah BUKU dipinjam oleh satu PEMINJAM**. Penyusunan tabel entity pada hubungan ini akan ditentukan oleh derajat hubungan dan jenis partisipasi anggota entity.

1. **Partisipasi Wajib Pada Kedua Entity**

   Hubungan ini terjadi jika aturan peminjaman yang berlaku adalah : **satu PEMINJAM perpustakaan <u>wajib</u> meminjam satu BUKU, dan satu buah BUKU <u>wajib</u> dipinjam oleh satu PEMINJAM**. Gambar E-R diagramnya sebagai berikut :

PEMINJAM(kodeanggota#, nama, alamat)
BUKU(kodebuku#, judul, pengarang)

Contoh keadaan isi dari entity PEMINJAM dan BUKU dihubungkan dengan penghubung MEMINJAM bisa digambarkan sebagai berikut :



Sehingga setelah melihat kondisi isi entitas diatas, maka hubungan kedua entitas tersebut direpresentasikan dalam <u>satu</u>

tabel saja. Hubungan kedua entity tersebut dilakukan dengan menggabungkan semua atribut PEMINJAM dan BUKU dalam satu tabel seperti contoh berikut :

Kodeanggota#	nama	alamat	Kodebuku#	judul	pengarang
A0001	Adi	Jl. Solo	B0002	Basis Data I	Ir. Abdul Kadir
A0002	Widhi	Jl. Mangga	B0001	Basis Data II	Ir. Indra
A0003	Santi	Jl. Jeruk	B0003	Pascal	Ir. Sayekti

Dengan demikian dapat dirumuskan :

*Pada **hubungan 1:1** dengan **partisipasi wajib kedua sisi** maka hubungan kedua entity direpresentasikan dalam **satu tabel yang mengandung seluruh atribut kedua entity**.*

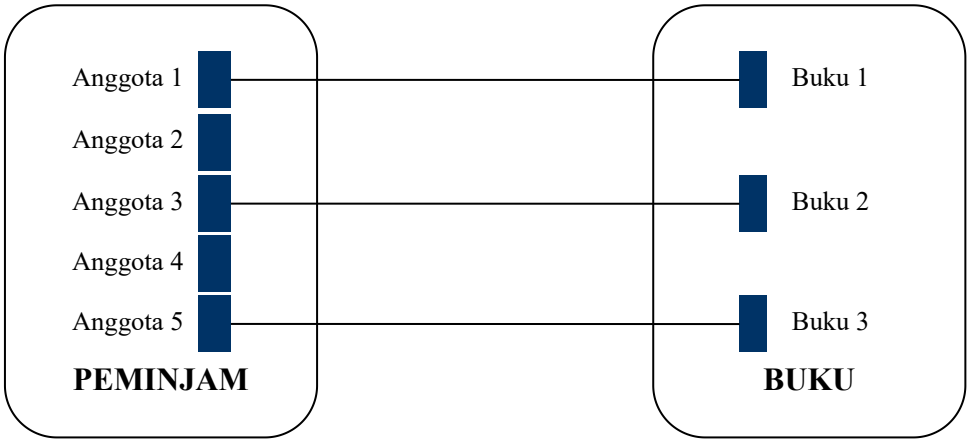
2. Partisipasi Wajib Hanya Pada Salah Satu Sisi Entity

Hubungan ini terjadi jika aturan peminjaman yang berlaku adalah : **satu** PEMINJAM perpustakaan **tidak wajib** meminjam satu BUKU, dan **satu** buah BUKU **wajib** dipinjam oleh satu PEMINJAM. Gambar E-R diagramnya sebagai berikut :

PEMINJAM(kodeanggota#, nama, alamat)

BUKU(kodebuku#, judul, pengarang)

Contoh keadaan isi dari entity PEMINJAM dan BUKU dihubungkan dengan penghubung MEMINJAM bisa digambarkan sebagai berikut :



Jika menggunakan cara yang sama seperti pada **partisi wajib dua sisi entity** maka bisa dihasilkan tabel sebagai berikut (contoh):

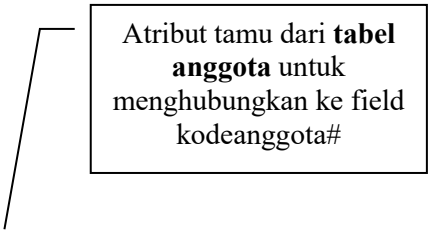
Kodeanggota#	**nama**	**alamat**	**Kodebuku#**	**judul**	**Pengarang**
A0001	Adi	Jl. Solo	B0001	Basis Data I	Ir. Abdul Kadir
A0002	Yudi	Jl. Yogya	-	-	-
A0003	Widhi	Jl. Mangga	B0002	Basis Data II	Ir. Indra
A0004	Arri	Jl. Sapen	-	-	-
A0005	Santi	Jl. Jeruk	B0003	Pascal	Ir. Sayekti

Ternyata pada kasus ini bisa berkecenderungan menghasilkan tabel dengan beberapa nilai atribut **kosong**. Untuk menghindari terjadinya nilai kosong maka hubungan 1:1 dengan partisipasi wajib satu sisi entity harus direpresentasikan dalam dua tabel. Hubungan

kedua entity tersebut dilakukan dengan membuat dua tabel dengan ketentuan : pada entity yang partisipasinya tidak wajib (PEMINJAM) dibuatkan tabel sesuai atribut PEMINJAM, sedangkan pada entity dengan partisipasi wajib BUKU membentuk tabel yang mengandung semua atribut BUKU ditambah **atribut tamu/kunci tamu** yang berasal dari kunci tabel entity PEMINJAM seperti contoh berikut :

PEMINJAM

Kodeanggota#	nama	alamat
A0001	Adi	Jl. Solo
A0002	Widhi	Jl. Mangga
A0003	Santi	Jl. Jeruk

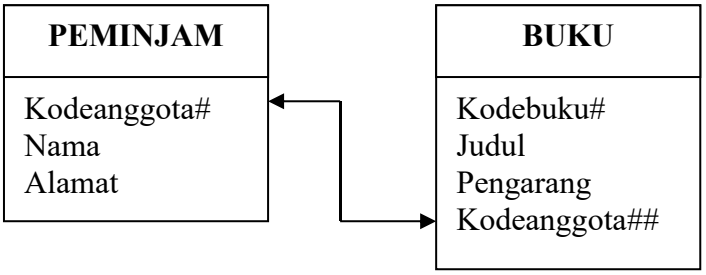


BUKU

Kodebuku#	judul	pengarang	Kodeanggota#
B0001	Basis Data II	Ir. Indra	A0002
B0002	Basis Data I	Ir. Abdul Kadir	A0001
B0003	Pascal	Ir. Sayekti	A0003

Dalam kasus ini atribut kodeanggota# menjadi atribut tamu pada entity BUKU dan digunakan sebagai penghubung MEMINJAM. Dengan demikian dapat dihindarkan adanya nilai kosong baik pada tabel PEMINJAM maupun BUKU.

Gambaran relasi antara tabel PEMINJAM dengan BUKU adalah sebagai berikut :



Field kunci tamu di tabel buku adalah kodeanggota# . Fungsi field ini adalah untuk menghubungkan antara entity BUKU dengan entity PEMINJAM dengan hubungan MEMINJAM.

Dengan demikian dapat dirumuskan :

*Pada **hubungan 1:1** dengan **partisipasi wajib salah satu sisi** maka hubungan kedua entity direpresentasikan dalam **dua tabel** dimana pada tabel yang memiliki **tingkat partisipasi wajib** akan memiliki **atribut tamu** berasal dari atribut kunci entity sisi lain untuk berfungsi **sebagai penghubung**.*

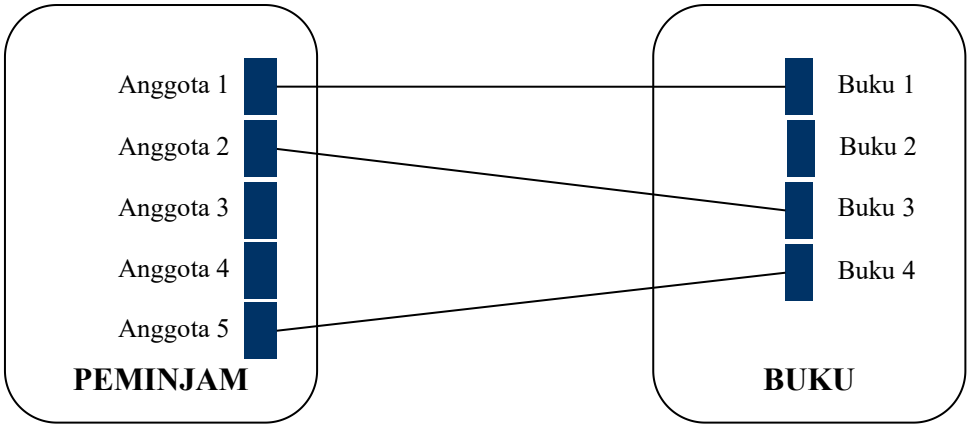
3. Partisipasi Tidak Wajib Pada Kedua Sisi

Hubungan ini terjadi jika aturan peminjaman yang berlaku adalah : **satu** PEMINJAM perpustakaan **tidak wajib** meminjam satu BUKU, dan satu buah BUKU **tidak wajib** dipinjam oleh satu PEMINJAM. Gambar E-R diagramnya sebagai berikut :

PEMINJAM(kodeanggota#, nama, alamat)

BUKU(kodebuku#, judul, pengarang)

Contoh keadaan isi dari entity PEMINJAM dan BUKU dihubungkan dengan penghubung MEMINJAM sesuai E-R diagram diatas bisa digambarkan sebagai berikut :



Pada bentuk diatas akan menghasilkan dua buah tabel yaitu tabel/entitas PEMINJAM dan tabel/entitas BUKU dengan contoh :

PEMINJAM

Kodeanggota#	nama	alamat
A0001	Adi	Jl. Solo
A0002	Yudi	Jl. Yogya
A0003	Widhi	Jl. Mangga
A0004	Arri	Jl. Sapen
A0005	Santi	Jl. Jeruk

BUKU

Kodebuku#	judul	pengarang
B0001	Basis Data II	Ir. Indra
B0002	Basis Data I	Ir. Abdul Kadir
B0003	Pascal	Ir. Sayekti

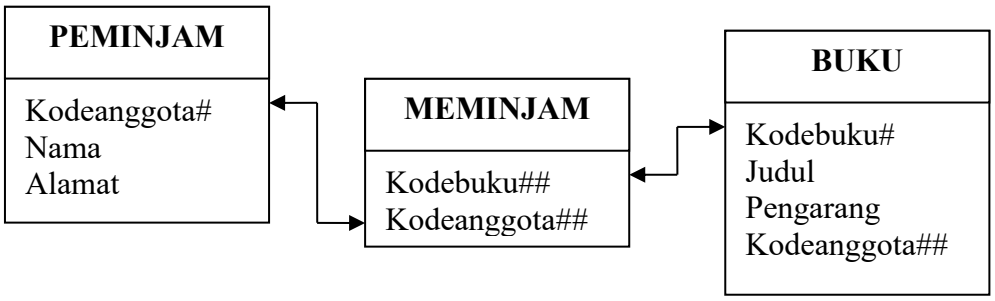
Serta sebuah tabel tambahan untuk mencatat hubungan MEMINJAM yang minimal berisi atribut kunci kedua tabel

MEMINJAM

Kodebuku#	Kodeanggota#
B0002	A0001
B0001	A0002
B0003	A0003

Dalam kasus ini atribut kodeanggota# dari PEMINJAM menjadi atribut tamu pada entity MEMINJAM dan atribut kodebuku# dari BUKU menjadi atribut tamu pada entity MEMINJAM. Entity/tabel baru MEMINJAM menjadi penghubung antara entity PEMINJAM dan BUKU. Dengan cara seperti ini maka dapat dihindarkan adanya nilai kosong baik pada tabel PEMINJAM maupun BUKU. Tabel baru MEMINJAM paling tidak harus mengandung atribut kunci dari tabel-tabel yang dihubungkan.

Gambar relasi antara tabel PEMINJAM dengan BUKU adalah sebagai berikut :



Dengan demikian dapat dirumuskan :

*Pada **hubungan 1:1** dengan **partisipasi tidak wajib kedua sisi** maka hubungan kedua entity direpresentasikan dalam **tiga tabel** dimana pada tabel ketiga akan memiliki **atribut tamu** berasal dari atribut kunci tabel-tabel lain yang dihubungkan karena berfungsi **sebagai penghubung** antar entity.*

Representasi Hubungan 1:m

Misalnya suatu perpustakaan memberikan aturan peminjaman bagi PEMINJAM terhadap BUKU. Aturan yang berlaku adalah : satu PEMINJAM perpustakaan meminjam satu atau beberapa buah buku BUKU, dan satu buah BUKU dipinjam oleh satu PEMINJAM (hubungan 1: m) . Gambar E-R diagramnya sebagai berikut :

PEMINJAM(kodeanggota#, nama, alamat)

BUKU(kodebuku#, judul, pengarang)

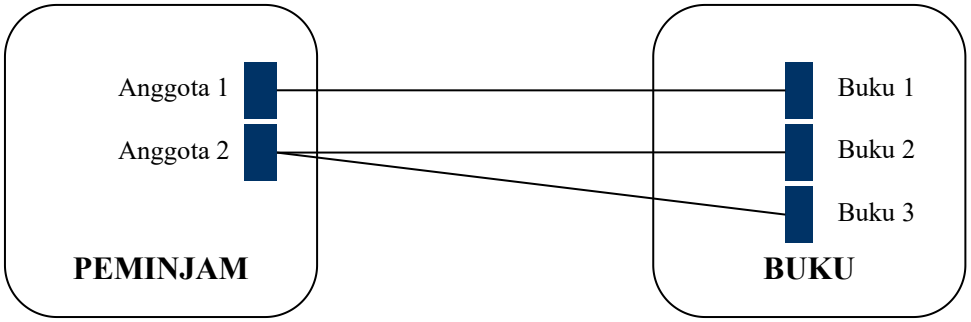
Dengan kondisi seperti di atas maka partisipasi di sisi 1 (PEMINJAM) tidak perlu diperhatikan, tetapi jenis partisipasi di sisi m (BUKU) akan menentukan penyajian tabel entity yang terbentuk.

1. **Partisipasi Wajib Pada Sisi "m"**

   Misalnya pada hubungan PEMINJAM dan BUKU ditentukan bahwa satu PEMINJAM perpustakaan meminjam satu atau beberapa buah buku BUKU, dan satu buah BUKU **wajib** dipinjam oleh satu PEMINJAM. Maka E-R diagram yang terbentuk menjadi sebagai berikut :

PEMINJAM(kodeanggota#, nama, alamat)
BUKU(kodebuku#, judul, pengarang)

Contoh keadaan isi dari entity PEMINJAM dan BUKU dihubungkan dengan penghubung MEMINJAM sesuai E-R diagram diatas bisa digambarkan sebagai berikut :



Pada bentuk diatas akan menghasilkan dua buah tabel yaitu tabel/entitas PEMINJAM dan tabel/entitas BUKU dengan contoh :

PEMINJAM

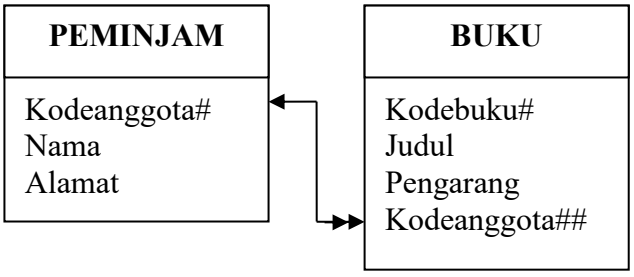
Kodeanggota#	nama	alamat
A0001	Adi	Jl. Solo
A0002	Widhi	Jl. Mangga

Atribut tamu dari **tabel anggota** untuk menghubungkan ke field kodeanggota#

BUKU

Kodebuku#	judul	pengarang	Kodeanggota#
B0001	Basis Data II	Ir. Indra	A0001
B0002	Basis Data I	Ir. Abdul Kadir	A0002
B0003	Pascal	Ir. Sayekti	A0002

Gambar relasi antara tabel PEMINJAM dengan BUKU adalah sebagai berikut :



Dengan demikian dapat dirumuskan :

*Pada **hubungan 1:m** dengan **partisipasi wajib sisi m** maka hubungan kedua entity direpresentasikan dalam **dua tabel** dimana pada tabel kedua akan memiliki **atribut tamu***

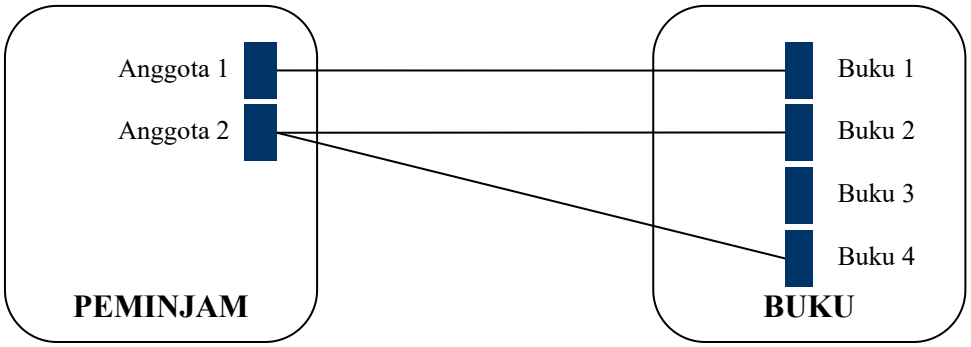
berasal dari atribut kunci tabel sisi 1 karena berfungsi ***sebagai penghubung*** *antar entity.*

2. Partisipasi Tidak Wajib Pada Sisi "m"

Misalnya pada hubungan PEMINJAM dan BUKU ditentukan bahwa satu PEMINJAM perpustakaan meminjam satu atau beberapa buah buku BUKU, dan satu buah BUKU **tidak wajib** dipinjam oleh satu PEMINJAM. Maka E-R diagram yang terbentuk menjadi sebagai berikut :

PEMINJAM(kodeanggota#, nama, alamat)
BUKU(kodebuku#, judul, pengarang)

Contoh keadaan isi dari entity PEMINJAM dan BUKU dihubungkan dengan penghubung MEMINJAM sesuai E-R diagram diatas bisa digambarkan sebagai berikut :



Pada bentuk diatas akan menghasilkan dua buah tabel yaitu tabel/entitas PEMINJAM dan tabel/entitas BUKU dengan contoh :

PEMINJAM

Kodeanggota#	nama	alamat
A0001	Adi	Jl. Solo
A0002	Yudi	Jl. Yogya

BUKU

Kodebuku#	judul	pengarang
B0001	Basis Data II	Ir. Indra
B0002	Basis Data I	Ir. Abdul Kadir
B0003	Pascal	Ir. Sayekti
B0004	Cobol	Ir. Andi

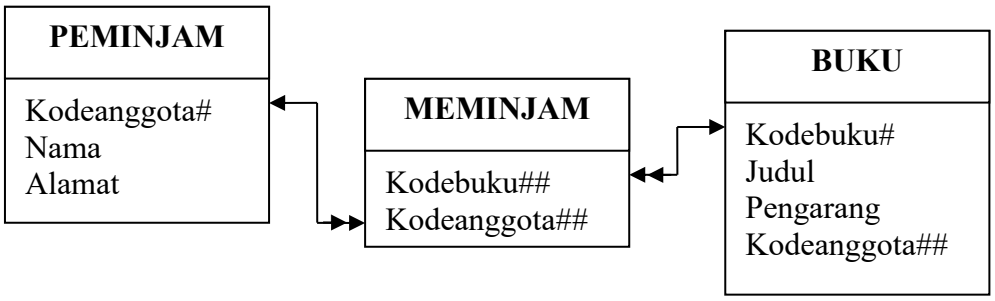
Serta sebuah tabel tambahan untuk mencatat hubungan MEMINJAM yang minimal berisi atribut kunci kedua tabel

MEMINJAM

Kodebuku#	Kodeanggota#
B0001	A0001
B0002	A0002
B0004	A0002

Dalam kasus ini atribut kodeanggota# dari PEMINJAM menjadi atribut tamu pada entity MEMINJAM dan atribut kodebuku# dari BUKU menjadi atribut tamu pada entity MEMINJAM. Entity/tabel baru MEMINJAM menjadi penghubung antara entity PEMINJAM dan BUKU. Dengan cara seperti ini maka dapat dihindarkan adanya nilai kosong baik pada tabel PEMINJAM maupun BUKU. Tabel baru MEMINJAM paling tidak harus mengandung atribut kunci dari tabel-tabel yang dihubungkan.

Gambaran relasi antara tabel PEMINJAM dengan BUKU adalah sebagai berikut :



Dengan demikian dapat dirumuskan :

*Pada **hubungan 1:1** dengan **partisipasi tidak wajib sisi m** maka hubungan kedua entity direpresentasikan dalam **tiga tabel** dimana pada tabel ketiga akan memiliki **atribut tamu** berasal dari atribut kunci tabel-tabel lain yang dihubungkan karena berfungsi **sebagai penghubung** antar entity.*

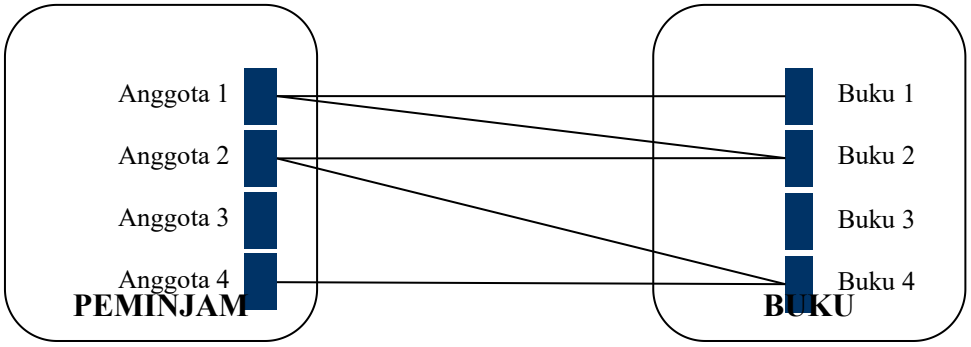
Representasi Hubungan m:n

Misalnya suatu perpustakaan memberikan aturan peminjaman bagi PEMINJAM terhadap BUKU : satu PEMINJAM perpustakaan boleh meminjam satu atau beberapa buah buku BUKU, dan satu buah BUKU bisa dipinjam oleh satu atau beberapa orang PEMINJAM (hubungan m:n) . Gambar E-R diagramnya sebagai berikut :

PEMINJAM(kodeanggota#, nama, alamat)

BUKU(kodebuku#, judul, pengarang)

Dengan kondisi seperti di atas maka partisipasi di kedua sisi (PEMINJAM dan BUKU) tidak perlu diperhatikan. Contoh keadaan isi dari entity PEMINJAM dan BUKU dihubungkan dengan penghubung MEMINJAM sesuai E-R diagram diatas bisa digambarkan sebagai berikut :



Pada bentuk diatas akan menghasilkan dua buah tabel yaitu tabel/entitas PEMINJAM dan tabel/entitas BUKU dengan contoh :

PEMINJAM

Kodeanggota#	nama	Alamat
A0001	Adi	Jl. Solo
A0002	Yudi	Jl. Yogya
A0003	Sinta	Jl. Jeruk
A0004	Widhi	Jl. Soga

BUKU

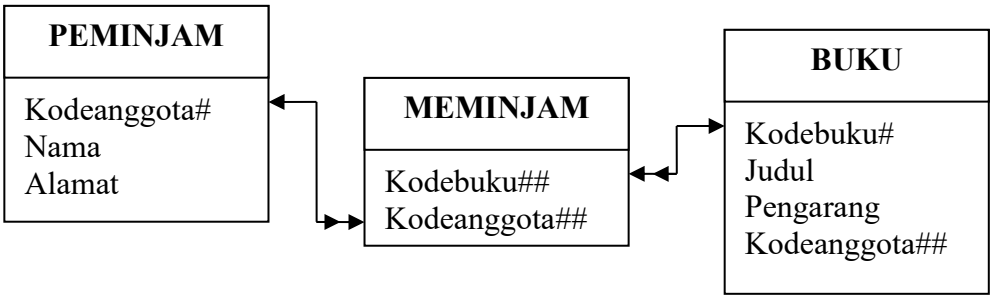
Kodebuku#	judul	pengarang
B0001	Basis Data II	Ir. Indra
B0002	Basis Data I	Ir. Abdul Kadir
B0003	Pascal	Ir. Sayekti
B0004	Cobol	Ir. Andi

Serta sebuah tabel tambahan untuk mencatat hubungan MEMINJAM yang minimal berisi atribut kunci kedua tabel

MEMINJAM

Kodebuku#	Kodeanggota#
B0001	A0001
B0002	A0001
B0002	A0002
B0004	A0002
B0004	A0004

Atribut kodeanggota# dari PEMINJAM menjadi atribut tamu pada MEMINJAM dan atribut kodebuku# dari BUKU juga menjadi atribut tamu pada MEMINJAM. Entity/tabel baru MEMINJAM menjadi penghubung antara PEMINJAM dan BUKU. Dengan cara seperti ini maka dapat dihindarkan adanya nilai kosong baik pada tabel PEMINJAM maupun BUKU. Tabel baru MEMINJAM paling tidak harus mengandung atribut kunci dari tabel-tabel yang dihubungkan. Gambaran relasi antara tabel PEMINJAM dengan BUKU adalah sebagai berikut :



Dengan demikian dapat dirumuskan :

*Pada **hubungan m:n** maka hubungan kedua entity direpresentasikan dalam **tiga tabel** dimana pada tabel ketiga akan memiliki **atribut tamu** berasal dari atribut kunci tabel-tabel lain yang dihubungkan karena berfungsi **sebagai penghubung** antar entity.*

Pelengkapan Atribut menuju Sistem Database Sempurna

Tabel-tabel yang dihasilkan oleh hasil konversi umumnya hanya menyebutkan atribut-atribut pokok saja. Penambahan atribut

baru untuk melengkapi diperbolehkan asalkan selalu berpedoman dalam bentuk normal yaitu dengan ketentuan :

1. Tidak boleh ada data ganda (repeating group).
2. Setiap determinan/field kunci dalam tabel dapat dipilih sebagai identitas tabel. Karena dalam penulisan kerangka tabel sudah dicantumkan identitas tabel, maka tidak diperbolehkan membuat identitas tabel baru.
3. Penambahan atribut hanyalah atribut yang akan bergantung penuh pada field kunci tabel.

Contoh :

Hasil dari konversi hubungan m:n adalah tabel berikut :

PEMINJAM

Kodeanggota#	nama	alamat
A0001	Adi	Jl. Solo
A0002	Yudi	Jl. Yogya

BUKU

Kodebuku#	Judul	pengarang
B0001	Basis Data II	Ir. Indra
B0002	Basis Data I	Ir. Abdul Kadir
B0003	Pascal	Ir. Sayekti
B0004	Cobol	Ir. Andi

MEMINJAM

Kodebuku#	Kodeanggota#
B0001	A0001
B0002	A0002
B0004	A0002

Maka kerangka tabelnya adalah sebagai berikut :

PEMINJAM(Kodeanggota#, nama, alamat)

BUKU(Kodebuku#, judul, pengarang)

MEMINJAM(Kodebuku#, Kodeanggota#)

Bisa dikembangkan/disempurnakan menjadi :

PEMINJAM(Kodeanggota#, nama, alamat, kota, no_ktp)
BUKU(Kodebuku#, judul, pengarang, penerbit)
MEMINJAM(Kodebuku#,Kodeanggota#,Tglpinjam,Tglkembali)

--- 000 ---

BAB VII
BASIS DATA SEBAGAI KOMPONEN SISTEM INFORMASI

A. Instalasi

Proses instalasi basis data pada sistem komputer dilakukan setelah Database Management System (DBMS) pembuatnya terpasang pada komputer tersebut. Hal-hal yang mempengaruhi proses instalasi adalah :

1. Sistem operasi yang dipakai

Jenis istem operasi tertentu hanya mendukung jenis basis data tertentu pula. Sistem operasi DOS hanya mendukung basis data Dbase, FoxBase, FoxPro yang berjalan di lingkungan DOS. Sistem Windows sebagai pengembangan sistem DOS mampu mendukung database pada sistem operasi DOS ditambah database dari aplikasi-aplikasi Windows.

2. Sisa ruang harddisk sebagai lokasi instalasi

Sisa ruang harddisk menentukan ukuran dari database yang hendak dipasang hingga dipergunakan. Semakin besar ukurannya maka database akan semakin mampu menampung lebih banyak data.

3. Kecepatan processor dan kapasitas memory pemroses

Kecepatan processor dan ketersediaan memory kerja merupakan komponen penentu unjuk kerja pemrosesan data dalam database. Semakin tinggi kecepatan kinerja processor, maka

semakin cepat proses pengolahan data. Semakin besar ukuran memory kerja maka semakin banyak data yang bisa ditampung sementara dalam memory untuk diolah oleh processor.

4. Jumlah pengguna

Basis data dengan satu pengguna (single user) akan diproses oleh satu processor saja sehingga tidak membebani sistem. Tetapi jika basis data dipaki pada lingkungan dengan banyak pengguna (multiuser) maka diperlukan perlakuan khusus supaya seluruh pengguna mendapatkan layanan data secara optimal dan sesuai haknya.

B. Pengamanan

Basis data pada sistem terkomputerisasi umumnya akan diakses oleh banyak pengguna secara bersamaan melalui sarana jaringan komputer. Dengan demikian data dapat diakses dari mana saja sehingga memberikan manfaat yang banyak bagi sistem. Namun hal ini juga membuka peluang bahwa data yang penting bisa diakses oleh siapa saja. Untuk mengantisipasi terjadinya hal ini maka umumnya DBMS menyediakan sarana proteksi data.

Dalam rangka untuk melindungi basis data beserta isinya maka pengamanan basis data sebenarnya dapat dilakukan dengan berbagai level.

1. Level Fisis

Pengamanan dilakukan dengan menempatkan sistem komputer pada ruang yang secara fisik tidak dapat diakses oleh sembarang orang. Misalnya meletakkan sistem komputer dalam

ruangan terkunci dan pengunjungnya pun terseleksi. Contoh kongkret pada pengamanan sistem komputer untuk KPU (Komisi Pemilihan Umum) yang terpisah dari sistem lain, dengan pengguna yang tertentu pula. Termasuk juga sistem komputer untuk penerimaan dan koreksi test CPNS (Calon Pegawai Negeri Sipil), dan sebagainya.

2. Level Manusia

Kewenangan pengguna harus diperhatikan dengan baik sehingga hanya orang yang berhak saka yang bisa mengakses data. Hal ini bisa dilakukan dengan memproteksi aplikasi dengan password. Hanya orang-orang yang mengetahui password saja yang bisa menggunakan basis data.

3. Level Sistem Operasi

Pada sistem jaringan seseorang dapat mengakses data dari jarak jauh (jauh dari pusat data), keamanan dalam level sistem operasi juga harus diperhatikan. Misalnya sistem operasi akan mengatur agar seseorang hanya bisa menggunakan komputer pada jam-jam tertentu, serta mengakses file-file tertentu saja sesuai haknya.

4. Level Sistem Basis Data

Pada level basis data terdapat pula pengaturan-pengaturan yang memungkinkan seseorang hanya dapat membaca data tertentu. Pada level ini pengaturan tersebut diatur melalui DBMS.

5. Gabungan beberapa atau seluruh level diatas.

Berkaitan dengan sistem control pengamanan, dalam suatu DBMS sering dijumpai istilah *privilege* (hak istimewa) atau *authority* (wewenang). Setiap pengguna mempunyai hak yang berbeda-beda dalam mengakses basis data. Contohnya, pengguna yang menangani penggajian memiliki password untuk melihat data gaji seseorang, tetapi tidak demikian halnya pengguna pada bagian pemasaran.

C. Integritas

Integritas adalah keakurasian atau keabsahan data dalam basis data. Untuk mendukung integritas, DBMS menyediakan kekangan integritas. Kegunaannya adalah untuk menjaga keabsahan data dalam basis data. Dengan adanya komponen integritas, kesalahan data dalam basis data dapat dideteksi dengan cepat.

Untuk mendukung integritas DBMS memiliki bagian yang disebut subsistem integritas. Subsistem ini bertanggungjawab terhadap pemantauan operasi pengubahan oleh pengguna (penyisipan, pemodifikasian, dan sebagainya) untuk menjamin operasi bersangkutan tidak melanggar aturan-aturan integritas yang telah diterapkan. Sebagai contoh, jika ada sebuah tabel memiliki kunci primer, maka ada jaminan dari sistem bahwa tabel tersebut tidak mungkin berisi data kembar. Tabel haruis bisa menolah penambahan record jika record baru mengakibatkan munculnya kunci kembar.

D. Konkurensi

Konkurensi adalah kemampuan dari basis data untuk diakses oleh banyak user dalam waktu yang bersamaan. Hal ini menjadi hal penting bagi DBMS yang diorientasikan untuk sistem multiuser.

--- 000 ---

BAB IX.
STRUCTURED QUERY LANGUAGE (SQL)

A. Pendahuluan

Structured Query Language (SQL) adalah suatu bahasa pemrograman standar yang dipergunakan untuk mengakses tabel-tabel dalam basis data sehingga bisa diatur sedemikian rupa sesuai dengan yang diinginkan. Perintah-perintah SQL terbagi menjadi beberapa macam, tetapi pada materi ini hanya akan dibahas dua macam saja yaitu :

- **Data Definition language** (perintah SQL untuk mendefinisikan sesuatu objek database)
- **Data Manipulation Language** (perintah SQL untuk memanipulasi suatu obyek database

SQL merupakan kumpulan perintah-perintah pemrograman yang dikhususkan bagi pengembang sistem atau pemakai sistem untuk melakukan tugas-tugas antara lain :

a. Memanggil data dari satu tabel atau lebih dengan mudah, singkat dan sederhana.
b. Memanipulasi data pada tabel-tabel dengan menyisipkan, menghapus, atau update record.
c. Mendapatkan ringkasan informasi mengenai data pada tabel seperti : total, jumlah record, nilai maksimum dan minimum dan nilai rata-rata.
d. Membuat, memodifikasi, atau menghapus tabel pada database.

e. Membuat dan meghapus indeks dari suatu database (hanya jenis database tertentu).
f. Dan lain sebagainya

B. Struktur Dasar Sintaks SQL

Pernyataan SQL setidaknya memiliki tiga bagian yaitu:

Deklarasi Parameter → bagian ini merupakan parameter optional yang diteruskan ke pernyataan SQL oleh program.

Pernyataan manipulatif → Bagian pernyataan ini memberitahu mesin query tindakan apa yang akan diambil, misalnya SELECT atau DELETE.

Deklarasi Opsi → memberitahu mesin query mengenai kondisi filter, pengelompokan data, atau pengurutan yang diterapkan pada data yang sedang diproses. Didalamnya termasuk klause WHERE, GROUP BY, dan ORDER BY.

Bagian-bagian itu disusun sebagai berikut :

[Deklarasi paremeter] Pernyataan manipulatif [Opsi]

Pada bab ini kebanyakan hanya menggunakan pernyataan manipulatif dan deklarasi opsi. Dengan menggunakan dua bagian pernyataan SQL ini, kita dapat membuat query-query untuk melaksanakan berbagai tugas. Pada tabel dibawah ini terdapat daftar beberapa klausa manipulatif dan fungsi tujuannya.

Tabel 1. Bagian-Bagian Pernyataan Manipulatif

Pernyataan Manipulatif	Fungsi
DELETE ... FROM	Menghapus record-record dari table
INSERT INTO	Menambahkan sekelompok record pada tabel
SELECT	Memanggil sekelompok record dan menempatkan record-record tersebut dalam suatu dynaset atau table
UPDATE	Memperbaharui nilai-nilai field dalam suatu tabel

Meskipun pernyataan-pernyataan manipulatif mengatakan pada mesin database apa yang perlu dilakukan, deklarasi opsi mengatakan juga field dan record mana yang diproses.

C. Data Definition Language

Pada bab ini akan dipelajari bagaimana mendefinisikan objek tabel. Materi ini akan diberikan langsung dengan contoh suatu tabel sederhana. Perhatikan Tabel di bawah ini :

Tabel Wisuda

No. Mahasiswa	Nama	Alamat	Tgl Lahir	Telepon	IPK	Keterangan

Anggaplah tabel diatas memiliki struktur sebagai berikut :

Tabel 2. Struktur Tabel Wisuda

No.	Field	Tipe	Ukuran	Keterangan
1.	NoMahasiswa	Char	8	Nomor mahasiswa
2.	Nama	Char	30	Nama mahasiswa
3.	Alamat	Char	50	Alamat mahasiswa
4.	TglLahir	Date	-	Tanggal lahir mahasiswa
5.	Telepon	Char	20	Telepon mahasiswa
6.	IPK	Numeric	-	Index Prestasi Kumulatif mahasiswa
7.	Keterangan	Memo	-	Keterangan kelulusan

Atau memiliki struktur pendek seperti berikut ini :

Wisuda(NoMahasiswa Char(30), Nama Char(50), Alamat Char(50), TglLahir Date, Telepon Char(20), IPK Numeric, Keterangan Memo)

Bagaimanakah mengimplementasikan perintah-perintah *Data Definition Language* untuk mendefinisikan tabel tersebut?

1. Membuat Tabel Wisuda

Perintah SQL untuk membuat tabel baru adalah CREATE TABLE

```
CREATE TABLE <nama table> (field1 tipedata1, field2 tipedata2, ..field n tipedata n)
```

Sebagai contoh, perintah SQL untuk membuat tabel diatas adalah sebagai berikut :

Create Table Wisuda (NoMahasiswa Char(30), Nama Char(50), Alamat Char(50), TglLahir Date, Telepon Char(20), IPK Numeric, Keterangan Memo)

Maka tabel diatas akan terbentuk dalam kondisi kosong, belum terisi data apapun seperti di bawah ini

No. Mahasiswa	Nama	Alamat	Tgl Lahir	Telepon	IPK	Keterangan

2. Menambah Field Baru Pada Tabel Wisuda

Perhatikan Tabel Wisuda yang sudah ditambah dengan field baru di bawah ini

Tabel Wisuda

No. Mahasiswa	Nama	Alamat	Tempat Lahir	Tgl Lahir	Telepon	IPK	Keterangan

Tabel diatas telah ditambah field baru yaitu "**TempatLahir**" yang bertipe **Char(20).** Perintah SQL untuk menambahkan field baru pada suatu tabel perintah SQL nya adalah :

```
ALTER TABLE <nama table> ADD (nama field & tipe data)
```

Sehingga perintah SQL untuk menambahkan field TempatLahir dengan tipe data Char(20) pada Tabel Wisuda adalah :

Alter Table Wisuda Add Tempatlahir Char(20)

Maka tabel Wisuda akan terwujud menjadi seperti diatas.

NB. Posisi/urutan field tidak berpengaruh terhadap nilai data dalam tabel. Sehingga dalam suatu tabel urutan posisi field bukanlah suatu hal yang penting.

3. Memodifikasi Field Yang Sudah Ada Pada Tabel Wisuda

Untuk memodifikasi struktur tabel yang sudah terbentuk dikenai perintah ALTER..MODIFY sebagai berikut :

```
ALTER TABLE <nama table> MODIFY (nama field dan modifikasinya)
```

Jika diinginkan field “TempatLahir” menjadi bertipe Memo maka perintah SQl untuk mengubahnya adalah sebagai berikut :

Alter Table Wisuda Modify Tempatlahir Memo

Maka field Tempatlahir pada tabel Wisuda berubah menjadi bertipe Memo.

NB. Sintaks SQL Modify tidak didukung oleh beberapa software termasuk oleh Microsoft Access 2002.

4. Menghapus Field Yang Sudah Ada Pada Tabel Wisuda

Perintah SQL untuk menghapus field yang tidak perlu juga menggunakan sintaks ALTER .. DROP seperti di bawah ini :

```
ALTER TABLE <nama tabel> DROP <nama field>
```

Jika field Tempatlahir tidak diinginkan lagi maka bisa dihapus dengan perintah SQL sebagai berikut :

Alter Table Wisuda Drop Tempatlahir

Maka bentuk tabel Wisuda kembali seperti semula.

5. Menghapus Sebuah Tabel

Menghapus sebuah tabel beserta isinya bisa dilakukan dengan perintah

```
DROP TABLE <nama tabel>
```

Apabila tabel Wisuda sudah benar-benar tidak diperlukan lagi maka bisa dihapus (beserta isi datanya) dengan perintah sebagai berikut :

Drop Table Wisuda

Maka tabel Wisuda benar-benar akan terhapus berikut beserta isinya

D. Data Manipulation Language

Pada bagian ini akan dipelajari bagaimana memanipulasi data dalam sebuah tabel. Materi ini akan diberikan langsung dengan contoh suatu tabel sederhana sebagai berikut .

Tabel Address

No.	Nama	Alamat	Phone	TglLahir	JmlAnak

Tabel Address diatas memiliki struktur sebagai berikut :

Address(No Numeric, Nama Char(30), Alamat Char(50), Phone Char(20), TglLahir Date, JmlAnak Numeric)

Bagaimanakah memanfaatkan sintaks-sintaks Data Manipulation Language untuk mengolah data dalam tabel diatas.

1. Memasukkan Data Dalam Tabel

Perintah dasar SQL untuk memasukkan/menyisipkan data dalam suatu tabel adalah sebagai berikut :

Cara 1. Memasukkan data untuk semua field lengkap.

```
INSERT INTO <table name> VALUES (<data1,data2,…data n>)
```

Misalnya, untuk mengisi data dalam tabel Address perintahnya :

```
INSERT INTO Address VALUES (1,'Acong','Yogya','523731','23/10/1944',4);
```

Data yang dituliskan harus sesuai dengan urutan field pada saat pendefinisiannya. Jika tipe data **char** dan **date** maka data harus diapit oleh tanda petik. Jika data berbentuk **Number/Numeric** tidak menggunakan tanda petik.

Cara 2. Memasukkan data untuk field-field yang diinginkan saja.

```
INSERT INTO <table name> (field1, field2,...field n) VALUES (<data1,data2,…data n>)
```

Misalnya, untuk mengisi data dalam tabel Address untuk field NO, NAMA dan ALAMAT saja perintahnya :

INSERT INTO Address (No, Nama, Alamat) VALUES (2,'Sitorus,'Klaten);
INSERT INTO Address (no, nama) VALUES (3,'Yonathan');

Contoh tabel Address secara lengkap setelah melalui dua proses pengisian :

No	Nama	Alamat	Phone	Tgllahir	Jmlanak
1	Acong	Yogya	523731	23/10/1944	4
2	Sitorus	Klaten			
3	Yonathan				

2. Menampilkan Data Dalam Tabel

Secara umum sintaks SQL untuk menampilkan data dalam tabel adalah menggunakan perintah SELECT. Bentuk umum perintah SELECT adalah sebagai berikut :

```
SELECT {DISTINCT | ALL { * | nama_kolom}…}
        FROM (nama_tabel)
  [WHERE kondisi]
  [GROUP BY nama_kolom]
  [HAVING kondisi]
  [ORDER BY]
```

Secara singkat untuk melihat data yang tersimpan dalam suatu tabel adalah dengan perintah SELECT sebagai berikut:

```
SELECT <fieldname1, fiieldname2,…> FROM <tablename>
```

Kita dapat menampilkan data field-field yang diinginkan dengan menentukannya pada bagian **fieldname**. Bila yang akan ditampilkan seluruh field kita bisa menggantikan **fieldname** dengan tanda *.

Misalnya :

SELECT No, Nama, Alamat, Phone, Tgllahir, Jmlanak
FROM Address;

Atau Perintah :

SELECT * FROM ADDRESS;

Hasil proses menampilkan isi tabel diatas adalah :

1	Acong	Yogya	523731	23/10/1944	4
2	Sitorus	Klaten			
3	Yonathan				

Kedua perintah diatas mempunyai maksud yang sama dan memberikan hasil yang sama pula. Operator logika dan matematika didalam SQL adalah :

Tabel 3. Operator Logika dan Matematika pada Sintaks SQL

Operator	Keterangan
=, >, <	Sama dengan, lebih dari, kurang dari
<> atau !=,>=,<=	Tidak sama dengan, lebih dan sama dengan, kurang dan sama dengan
And, Or, Not	Dan, atau, negasi

Operator-operator tersebut dapat digunakan pada perintah SELECT untuk menampilkan data-data tertentu saja. Untuk

melakukan hal ini didalam perintah SELECT ditambahkan perintah kondisi yaitu **WHERE**. Misalnya :

SELECT * FROM Address WHERE nama='Yonathan';

TANDA % & LIKE

Tanda % dapat digunakan untuk mempermudah pencarian data. Tanda harus diikuti operator **LIKE.** Misalnya :

SELECT * FROM Address WHERE UPPER(nama) LIKE '%on%';

Perintah ini akan menampilkan semua data yang dalam field **nama** mengandung huruf on

1	Acong	Yogya	523731	23/10/1944	4
3	Yonathan				

SELECT * FROM Address WHERE UPPER(nama) LIKE 'y%';

Perintah ini akan menampilkan semua data yang data pada field **nama** diawali huruf 'y'.

3	Yonathan				

OPERATOR BETWEEN

Operator ini digunakan untuk mencari data diantara dua data.

Misalnya :

SELECT * FROM Address WHERE no BETWEEN 2 AND 4;

OPERATOR IN

Digunakan untuk mencari data secara search list dari beberapa kondisi yang berbeda.

Misalnya :

SELECT * FROM Address WHERE UPPER(Alamat) IN ('Yogya','Klaten');

Jika ditambah klausa: **ORDER BY ALAMAT**, akan muncul berkelompok sesuai alamatnya.

3. Melakukan UPDATE Data.

Data dalam tabel bisa diperbaiki atau diperbaharui. Merubah data yang ada dalam tabel dapat dilakukan dengan perintah :

```
UPDATE <nama tabel> SET <fieldname1=newdata1,.. fieldname n=newdata n>
        WHERE <kondisi>;
```

Misalnya :

```
UPDATE ADDRESS SET alamat='Klaten',Phone='25445' WHERE no=2;
```

Perintah diatas akan memperbaiki data no=2 dengan perbaikan alamat adalah "Klaten" dan Phone adalah "25445".

4. Menghapus Data Dalam Tabel

Data yang tidak diperlukan bisa dihapus. Pengertian menghapus dalam hal ini adalah menghapus data dalam satuan record (baris). Untuk menghapus record-record data yang ada dalam tabel dilakukan dengan perintah :

DELETE FROM <nama tabel> WHERE <kondisi>;

Misalnya :

DELETE FROM Address WHERE UPPER(nama)='ACONG';

Perintah diatas akan menghapus record (baris) pertama yaitu record dengan nama=”ACONG”

---- 000 ----

DAFTAR PUSTAKA

Fathansyah, *Basis Data*, Informatika Bandung, Bandung, 2015.

Kadir, Abdul, *Konsep dan Tuntunan Praktis Basis Data*, Andi Offset, Yogyakarta, 2021.

Kristanto, Harianto, *Sistem Basis Data*, Andi Offset, Yogyakarta, 1998

Sutantha, Edhy, *Sistem Basis Data Konsep dan Peranannya dalam Sistem Informasi Manajemen*, Andi Offset, Yogyakarta, 1997

Waljiyanto, *Basis Data*, Penerbit J & J Learning, Yogyakarta, 2010

Kata Pengantar

Alhamdulillah, Diktat Basis Data telah selesai dibuat. Semoga bermanfaat bagi pembacanya.

Buku ini dirancang untuk selalu dinamis mengikuti trend perkembangan teknologi informasi. Oleh karena itu isi bahan ajar ini akan berbeda dengan bahan ajar pada versi-versi sebelumnya.

Bahan ajar ini dirancang sebagai panduan dalam perkuliahan dan bukanlah buku acuan yang memuat teori secara mendetil. Pada daftar pustaka terdapat daftar buku-buku induk yang diacu untuk membuat bahan ajar ini. Diharapkan mahasiswa telah secara mandiri mempelajari buku acuan dalam daftar pustaka tersebut.

Keberhasilan dalam perkuliahan sangat dipengaruhi oleh persiapan mahasiswa menjelang kuliah setiap harinya. Sekali lagi, mempelajari materi dan kepustakaan dalam buku ini sebelum kuliah sangatlah disarankan.

Demikian kata pengantar kami, semoga bermanfaat dan selamat belajar.

Tim Pengembang Materi

DAFTAR ISI