

BAB I**KONSEP DASAR DAN DEFINISI BASIS DATA****Tujuan :**

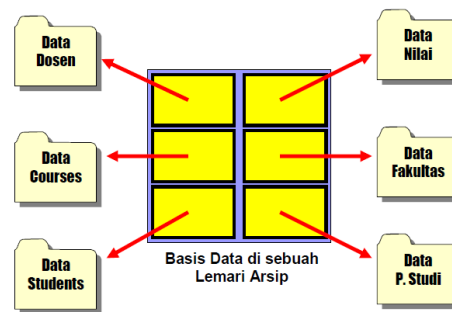
1. Mengetahui konsep dasar dan definisi basis data
2. Mengetahui komponen basis data
3. Mengetahui keuntungan dan kerugian basis data
4. Mengetahui penggunaan basis data

1.1 Konsep Dasar dan Definisi

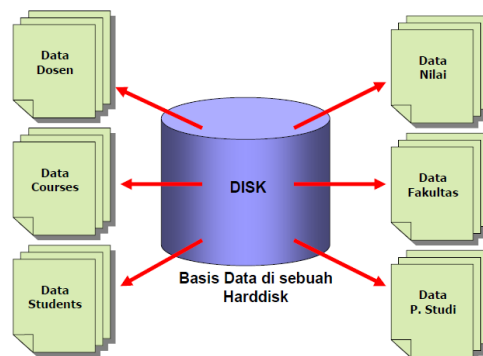
Sebenarnya basis data tidak selalu berhubungan dengan komputer karena pencatatan secara manual ke dalam suatu buku besar pun bisa dianggap sebagai suatu basis data, akan tetapi saat ini bila berbicara mengenai teknologi basis data maka akan selalu dihubungkan dengan teknologi komputer karena kedua hal tersebut berjalan beriringan.

Banyak definisi yang digunakan untuk menjelaskan istilah basis data yaitu :

- Himpunan kelompok data (arsip) yang saling berhubungan yang diorganisasikan sedemikian rupa agar kelak dapat dimanfaatkan kembali secara cepat dan mudah.
- Kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan (redundansi) yang tidak perlu, untuk memenuhi berbagai kebutuhan.
- Kumpulan file/tabel/arsip/dokumen yang saling berhubungan yang disimpan dalam media penyimpanan elektronik.
- Basis Data (*Database*), dapat dibayangkan sebagai sebuah lemari arsip. Jika kita memiliki lemari arsip dan berwenang/bertugas untuk mengelolanya, maka kemungkinan besar kita akan melakukan hal-hal seperti : memberi sampul/map pada kumpulan/bundel arsip yang akan disimpan, menentukan kelompok/jenis arsip, memberi nomor dengan pola tertentu yang nilainya unik pada setiap sampul/map.



Gambar 1. basis data di sebuah lemari arsip



Gambar 2. basis data di sebuah harddisk

1.2 Komponen Basis Data

1. Data (Data Operasional, Data Masukan, Data Keluaran)
2. Hardware (Perangkat Keras)
3. Software (Perangkat Lunak)
4. User / Pemakai

Keterangan :

1. Data

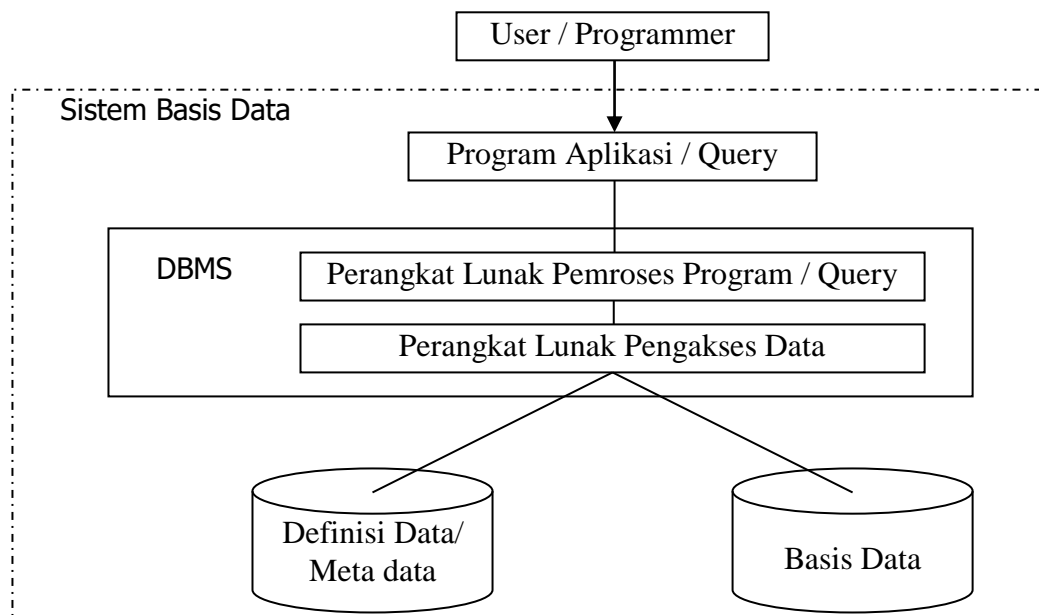
- **Data Operasional** : yaitu data yang disimpan dalam basis data baik itu berupa data master maupun data transaksi.
- **Data Masukan (Input data)** : yaitu data dari luar sistem yang dimasukkan melalui peralatan input seperti keyboard yang dapat mengubah data operasional.
- **Data Keluaran (Output data)** : yaitu data berupa laporan melalui peralatan output seperti screen, printer dll, sebagai hasil proses dari dalam suatu sistem yang mengakses data operasional.

2. **Hardware (Perangkat Keras)** : Terdiri dari semua peralatan komputer yang berbentuk fisik yang digunakan untuk pengelolaan basis data, seperti : HardDisk, Keyboard, Monitor dll.
3. **Software (Perangkat Lunak)** : Digunakan sebagai interface (antar Muka) antara pemakai dengan data fisik basis data, misalnya : Program-program aplikasi (POS, Inventory System dll) maupun DBMS (MySQL, SQL Server, Oracle dll).
4. **User atau Pemakai** dibagi menjadi 4 bagian :
 1. Naïve User / Pemakai Awam
Yaitu pemakai yang tidak berpengalaman / tidak mahir. Berinteraksi dengan basis data menggunakan program aplikasi yang khusus dibuat untuk suatu keperluan. Misal : Teller Bank, Kasir, Web User.
 2. Sophisticated User / Pengguna Canggih
Yaitu Pemakai yang berinteraksi dengan basis data tidak melalui program melainkan menggunakan bahasa query. Misal : Analis sistem, Pelaku End User Computing.
 3. Application Programmer / Pemrogram Aplikasi
Yaitu profesional dibidang komputer yang membuat program aplikasi menggunakan tool untuk membuat program aplikasi seperti Visual Foxpro, Delphi dll.
 4. Data Base Administrator (DBA)
Yaitu orang atau tim yang bertugas untuk mengelola sistem basis data secara keseluruhan. Tugas dari DBA antara lain :
 - a. Mendefinisikan maupun modifikasi Skema (dengan pernyataan DDL)
 - b. Mendefinisikan struktur tempat penyimpanan dan metode akses
 - c. Mendefinisikan Batasan Integritas
 - d. Pemberian Kewenangan I Hak Akses
 - e. Pemeliharaan (Backup, recovery, memantau kinerja)

1.3 Sistem Basis Data

Sistem Basis Data adalah gabungan antara Basis data dan perangkat lunak DBMS termasuk di dalamnya program aplikasi yang dibuat dan bekerja dalam

suatu sistem yang bertujuan untuk dapat memanipulasi data dari basis data sehingga diperoleh informasi yang diinginkan.



Gambar 3. Gambaran Lingkungan Sistem Basis Data

Terdapat perbedaan antara pendekatan pemrosesan berkas dan pendekatan pemrosesan basis data. Dalam pendekatan basis data sejumlah data disimpan dalam satu tempat dengan definisi data yang tetap sehingga dapat diakses oleh beberapa pemakai dengan berbagai program aplikasi. Definisi data disimpan dalam sistem katalog (terpisah dari data → *program-data independent*) yang berisi informasi tentang struktur, tipe dan format penyimpanan data. Informasi ini disebut meta data. Meta data digunakan oleh DBMS untuk mengakses data.

Pada pendekatan pemrosesan berkas, struktur data didefinisikan dalam program aplikasi sehingga hanya program aplikasi tersebut yang dapat mengakses datanya.

1.4 Keuntungan dan Kerugian Basis Data

Keuntungan pendekatan basis data

Berikut ini beberapa keuntungan menggunakan pendekatan basis data secara elektronis :

1. Kecepatan dan kemudahan (Speed)

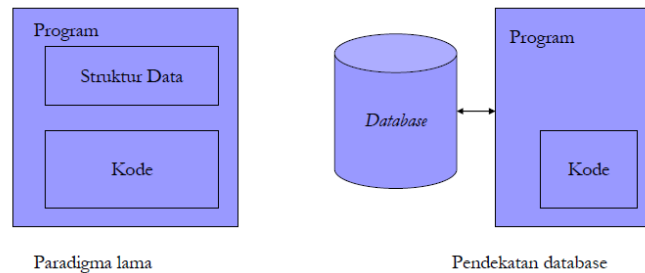
Mesin mampu mengambil dan mengolah informasi jauh lebih cepat dibandingkan dengan manusia.

2. Efisiensi ruang penyimpanan (Space)

Data dikodekan secara elektronik dan disimpan dalam sebuah media (Harddisk).

3. Independensi Program – Data

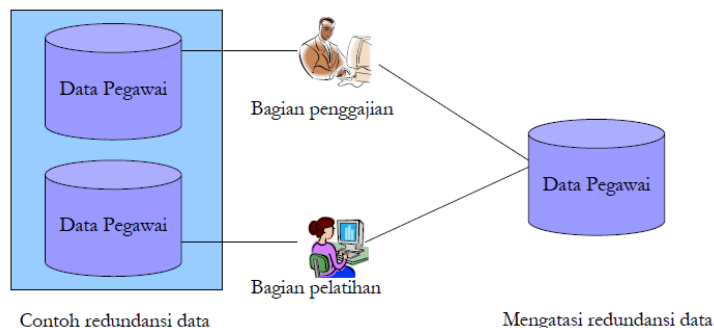
- Struktur data pada basis data terpisah dengan program
- Perubahan struktur data tidak membuat program harus dimodifikasi.



Gambar 4. independensi program - data

4. Meminimalkan redundansi data

- Redundansi data dapat dikurangi dengan cara data yang sama untuk aplikasi yang berbeda dijadikan satu.



Gambar 5. redundansi data

5. Meningkatkan Konsistensi Data

- Pengurangan redundansi data berimplikasi pada peningkatan konsistensi data (mengurangi kemungkinan untuk tidak konsisten).
- Contoh : Dua biro perjalanan tidak terhubung ke *database* (basis data) milik maskapai penerbangan. Apa yang terjadi kalau salah satu biro sudah menjual tempat duduk nomor 4 padahal biro yang lain tidak tahu?

6. Meningkatkan Kemampuan Berbagi data

- Data dapat diakses oleh banyak pemakai dengan tetap memperhatikan otorisasi.

- Istilah *multiuser* menyatakan bahwa sebuah data bisa diakses oleh banyak orang dalam waktu yang bersamaan.
7. Meningkatkan pencapaian standarisasi
 - Standarisasi seperti nama data, panjang data, kemungkinan nilainya, dan bahkan prosedur untuk mengaksesnya dapat diatur oleh yang berwenang (DBA). Contoh : Nama pegawai selalu betipe Alphanumeric dengan panjang maksimal 35 karakter. Semua pemrogram menggunakan standar tersebut.
 8. Meningkatkan kualitas data.
 - Kualitas data sangat berpengaruh terhadap pemerolehan informasi yang berkualitas.
 - Adanya kekangan (*constraint*) dalam *database* membuat pelanggaran terhadap isi data oleh pemakai tidak akan ditoleransi oleh sistem dengan sendirinya.
 - Kekangan adalah suatu aturan yang diterapkan pada data dan tidak bisa dilanggar oleh pemakai.
 - Contoh : Agama hanya bisa diisi dengan I, K, H, B, P. Sistem *database* akan menolak kalau huruf X dicoba untuk dimasukkan.
 9. Mengurangi pemeliharaan program.
 - Perubahan terhadap struktur data dengan berbagai alasan seringkali dilakukan selama tahapan pemeliharaan, misalnya data baru ditambahkan atau panjang suatu data ditambah. Perubahan seperti ini tidak selalu membuat program-program yang telah jadi harus ikut diubah.

Kerugian pemakaian sistem basis data

1. Mahal
 - Diperlukan hardware tambahan : CPU yang lebih besar, Terminal yang lebih banyak, Alat untuk komunikasi.
 - Biaya performance yang lebih besar : Listrik, Personil yang lebih tinggi klasifikasinya, Biaya telekomunikasi yang antar lokasi/kota.
2. Kompleks
3. Kesulitan dalam pemeliharaan (prosedur backup & recovery sulit).

1.5 Penggunaan Aplikasi Basis Data (*Database*)

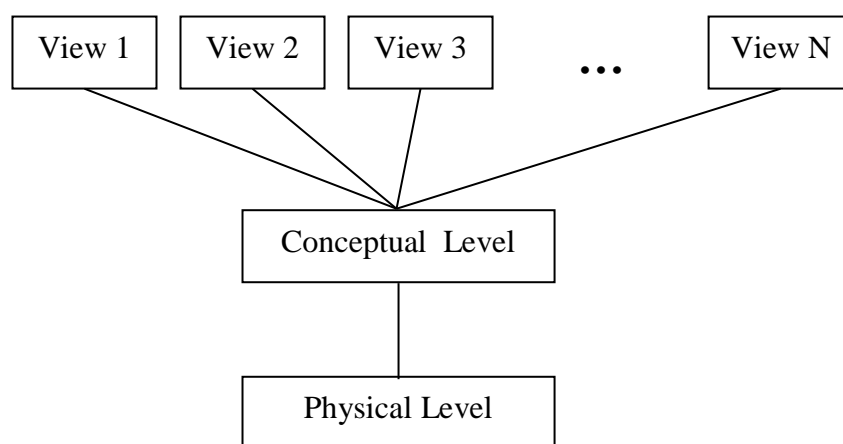
1. Perbankan : pengelolaan data nasabah, akunting, rekening, pinjaman, transaksi perbankan.
2. Penerbangan : pengelolaan data reservasi dan penjadwalan
3. Jasa Paket : tarif paket, tracking paket
4. Lembaga pendidikan (universitas/ perguruan tinggi) : pengelolaan pendaftaran, informasi mahasiswa, registrasi matakuliah, daftar nilai.
5. kepegawaian : pengelolaan data pegawai, riwayat pekerjaan, gaji
6. Telekomunikasi : tariff SLJJ, tagihan bulanan, saldo kartu Prabayar.
7. Toko buku : daftar buku, riwayat pembelian.
8. Penjualan : pengelolaan data customer, produk, penjualan
9. Pabrik : pengelolaan data produksi, persediaan barang, pemesanan, agen

1.6 Arsitektur Basis Data

Sesuai ANSI SPARC 1975, arsitektur untuk sistem basis data terdiri dari tiga tingkat yaitu:

1. Tingkat Internal atau fisik / Physical Level
2. Tingkat Konsep / Conceptual Level
3. Tingkat Pandangan / View Level / External Level

Digambarkan sebagai berikut :



Gambar 6. Arsitektur 3 tingkat

Tujuan adanya arsitektur 3 tingkat adalah memunculkan abstraksi dan memisahkan pandangan pemakai terhadap basis data dan cara implementasinya

secara fisik sehingga memperoleh keuntungan ketidakbergantungan / independensi data yang tinggi.

Alasan :

1. Tiap pemakai dapat mengakses data yang sama tetapi dengan pandangan yang berbeda, dengan kata lain user view sesuai kebutuhan dan bisa berbeda tiap user.
2. User tidak berurusan dengan kerumitan penyimpanan fisik basis data.
3. DBA dapat melakukan perubahan struktur penyimpanan basis data tanpa mempengaruhi user view.

Physical Level / Tingkat Internal / Tingkat Fisik

1. Merupakan tingkat terendah.
2. Mendeskripsikan cara penyimpanan fisik data.
3. Mengurusi alokasi ruang penyimpanan data dan indeks.
4. Berhubungan dengan metode manajemen file untuk menyimpan dan mengakses data.

Conceptual Level / Tingkat Konsep

1. Mendeskripsikan tentang data apa yang disimpan.
2. Menyatakan Entitas, Atribut dan keterhubungannya (relationships) serta konstrain maupun informasi mengenai keamanan dan integritas data.
3. Berisi Struktur Logik.

View Level / Tingkat Pandangan / External Level

1. Berinteraksi dengan sebagian dari basis data (sesuai kebutuhan user).
2. View / pandangan mendefinisikan satu bagian untuk satu kelompok pemakai tertentu.
3. Dapat menyediakan banyak pandangan berbeda pada basis data yang sama.

1.7 Ketakbergantungan / Independensi Data

Kemampuan memodifikasi skema di satu tingkat tidak mempengaruhi definisi skema di tingkat yang lain

Ada 2 macam independensi :

1. Logical data independence / ketidakbergantungan data secara logik
 - a. Perubahan skema konsep tak pengaruhi skema pandangan / program aplikasi.

- b. Misal : penambahan suatu atribut dimungkinkan tanpa harus menulis ulang program aplikasi.
 - c. Catatan : Perubahan skema konsep untuk aplikasi yang berhubungan ada kemungkinan tetap mempengaruhi tingkat pandangan, contohnya penambahan atribut golongan darah pada struktur data mahasiswa → agar dapat diisi dengan mudah maka program aplikasi yang sudah ada juga harus sedikit dirubah untuk fasilitas pengisian gol. Darah.
2. Physical data independence / Ketakbergantungan secara fisik
 - a. Perubahan skema fisik tak pengaruhi skema konsep.
 - b. Misal : Penambahan indeks seharusnya tidak akan mempengaruhi tingkat konsep maupun pandangan, hanya mungkin yang cukup terasa pada tingkat pandangan adalah peningkatan kinerja.

1.8 Model Data

Konsep yang digunakan untuk membuat struktur Basis data terdapat 2 kelompok model data :

1. Konseptual / high level
Bagaimana pemakai basis data memandang / memperlakukan data.
2. Fisikal / Low Level
Bagaimana deskripsi detail data disimpan dalam komputer (format perekaman, urutan dan indexing).

Model data konseptual

Terdiri dari 2 macam :

1. Object base data model (Model data berbasis object)
Menjelaskan hubungan logik antar data dalam suatu basis data berdasar obyek datanya.
Mis : Entity Relationship (E-R), Semantik, Object Oriented
2. Record Base data Model (Model Data berbasis Record)
Menjelaskan hubungan logik antar data berdasar record.
Digunakan untuk menguraikan implementasi sistem basis data.
Mis : Relational, Hirarki, Jaringan.

BAB II**KONSEPTUAL BASIS DATA DAN ENTITAS-RELASIONAL****Tujuan :**

1. Mengetahui beberapa teknik alternatif pemodelan data
2. Memahami konsep relasi
3. Mampu menggunakan simbol-simbol ERD

2.1 E-R Model

1. Entity Relationship Model → Dikenalkan oleh Chen (1976).
2. Berdasarkan anggapan bahwa dunia nyata terdiri dari koleksi obyek-obyek dasar yang dinamakan entitas (Entity) serta hubungan (Relationship) antara entitas-entitas tersebut.
3. Tak Bergantung DBMS dan platform perangkat keras.

Digunakan untuk :

1. Mengembangkan Model Konseptual
2. Menjelaskan Struktur Basis Data
3. Memberikan gambaran kepada pengguna terhadap data

Tiga Komponen Penting dalam Model E-R adalah : **Entity, Atribut dan Relasi**

ENTITY / ENTITAS

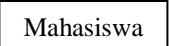
→ Obyek di dunia nyata yang dapat dibedakan dengan obyek yang lainnya Set Entity / Himpunan Entitas : Kumpulan Entity yang serupa.

Set Entity dapat berupa :

Fisik : Mahasiswa, Pasien, Kendaraan

Konsep / Logik : Pekerjaan, Mata Kuliah, Kursus

Simbol Entity :  (berupa kotak Segiempat)

Contoh Entity → Prabu, Fadil, Afif → dimodelkan sebagai 

ATRIBUT

→ Ciri atau karakteristik yang bermakna untuk mendeskripsikan entitas

→ Bertujuan untuk membedakan obyek-obyek di dalam entity

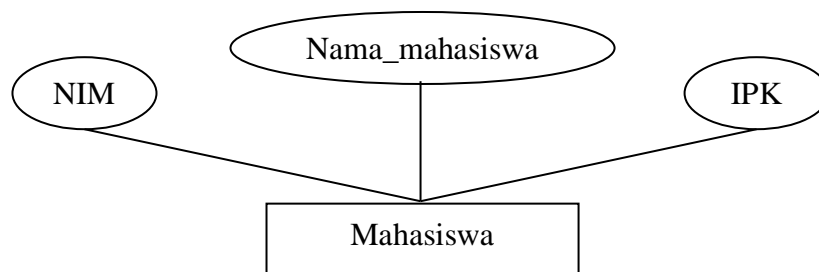
Karakteristik yang bermakna dan tidak bermakna

contoh untuk entity mahasiswa maka :

1. Bermakna : NIM, nama_mahasiswa, IPK
2. Tidak bermakna : tinggi_badan, berat badan

Tinggi_badan dan berat_badan menjadi karakteristik yang bermakna bagi entity petinju.

Simbol atribut : elips dan dihubungkan dengan entity melalui garis

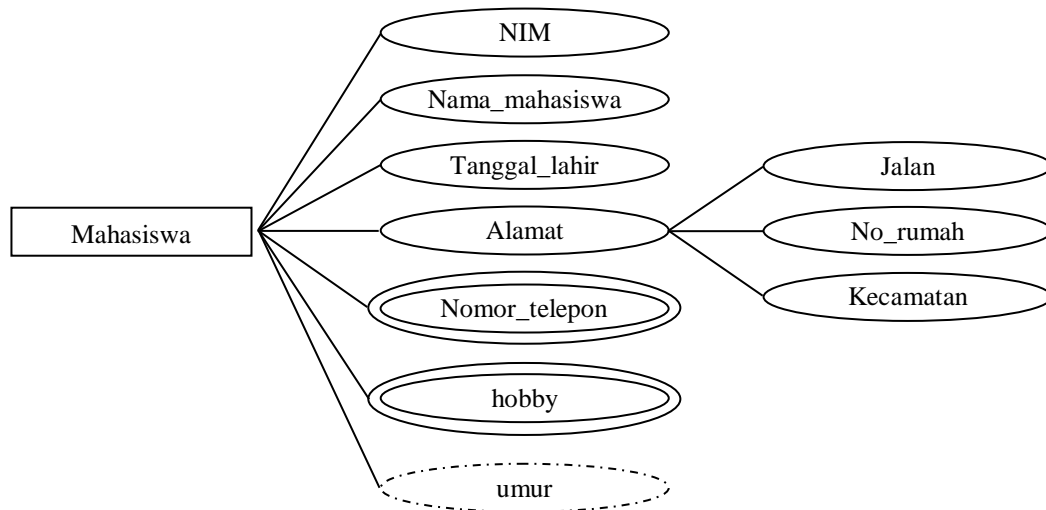


JENIS-JENIS ATRIBUT :

1. Atribut Sederhana
 - a. Atribut yang disusun hanya dari satu komponen tunggal dengan keberadaan bebas. Atribut sederhana tak dapat dibagi lagi → atribut atomik
 - b. Misal : nama, jenis_kelamin
2. Atribut Komposit
 - c. Atribut yang disusun dari banyak komponen yang masing-masing keberadaannya bebas. Atribut dapat dipecah menjadi komponen yang lebih kecil yang keberadaannya bebas.
 - d. Misal : alamat → jalan, no_rumah, kecamatan
3. Atribut Tunggal
 - a. Mengandung hanya satu nilai. Bisa terdiri dari atribut sederhana atau atribut komposit.
 - b. Misal : NIM, No_karyawan, Agama.
 - c. Ket : Seorang mahasiswa hanya punya 1 NIM, atau 1 Agama.

4. Atribut Bernilai Banyak / Jamak
 - a. Atribut yang mengandung banyak nilai.
 - b. Misal : hobby, ketrampilan, nomor_telepon.
 - c. Ket : Seorang mahasiswa bisa memiliki banyak hobby atau lebih dari 1 nomor_telepon.
5. Atribut turunan
 - a. Atribut yang mengandung nilai dimana nilai tersebut bisa diperoleh dari hasil kalkulasi atau penerapan algoritma ke atribut lain.
 - b. Misal : atribut umur → bisa diperoleh dari atribut tanggal_lahir.

Cara penggambaran beberapa jenis atribut di atas :



Gambar 6. Cara Penggambaran Atribut

Domain Atribut

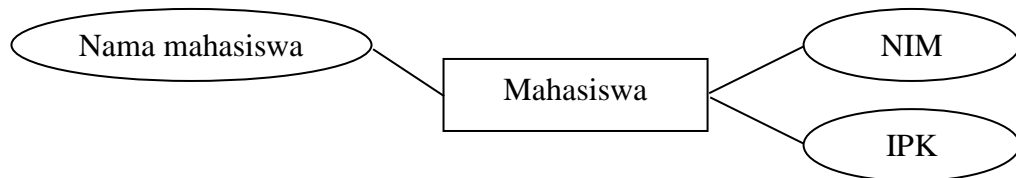
1. Himpunan nilai yang diberikan ke suatu atribut.
2. Termasuk diantaranya : tipe data, panjang, konstrain, format, dll
3. Atribut yang berbeda boleh mempunyai domain yang sama.

Key (Kunci)

1. Himpunan dari satu atau lebih atribut yang digunakan untuk membedakan antara satu entity dengan entity yang lain.
2. Harus unik dan tidak boleh kosong.
3. Superkey Candidate key, primary key.

4. Primary Key :

- Candidate key yang dipilih untuk identifikasi entitas pada himpunan entitas.
- Digambarkan dengan menambah garis bawah.

**RELASI**

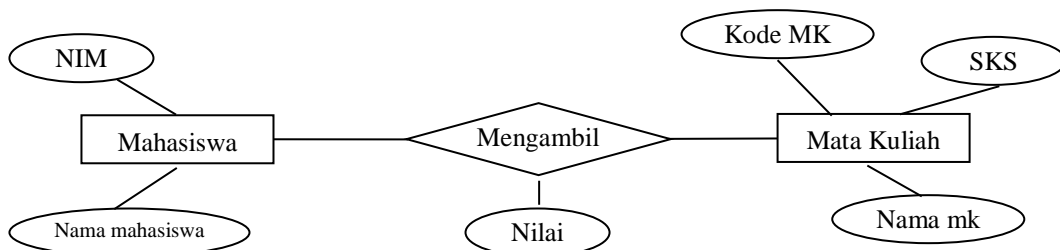
- Himpunan antara satu himpunan entitas dengan himpunan entitas yang lain.
- Dilambangkan dengan simbol diamond (wajik) yang di dalamnya terdapat kata kerja / kata hubung,

Misal :

Dosen Mengajar Mahasiswa

**Atribut Hubungan / atribut relasi**

- Uraian tentang suatu hubungan
- Berguna untuk menjelaskan suatu hubungan
- Jika atribut hubungan cukup banyak maka perlu dipertimbangkan untuk menjadi entity baru.

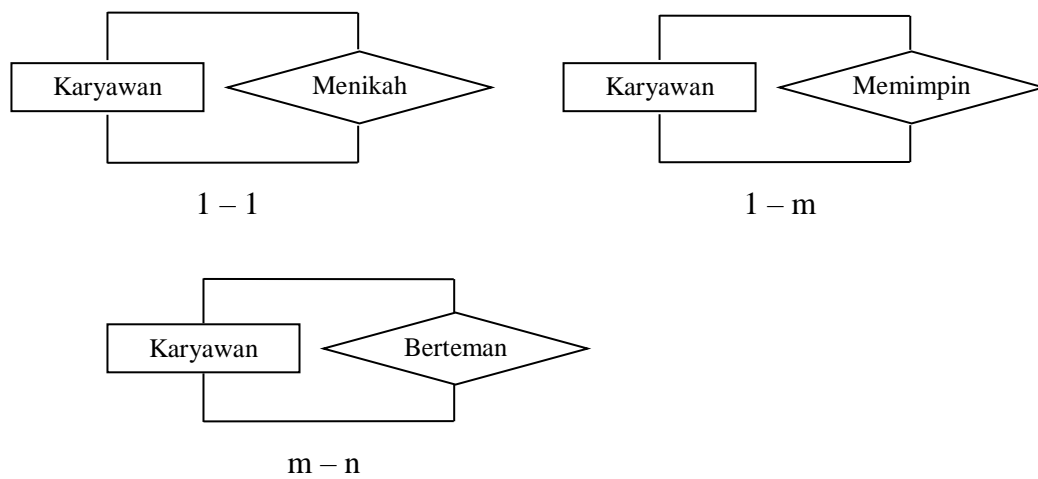
**Derajat Relasi :**

- Jumlah entitas yang berpartisipasi dalam suatu relasi

Ada tiga macam :

- Unary relationship / rekursif relationship

Hanya melibatkan 1 entitas / berderajat 1



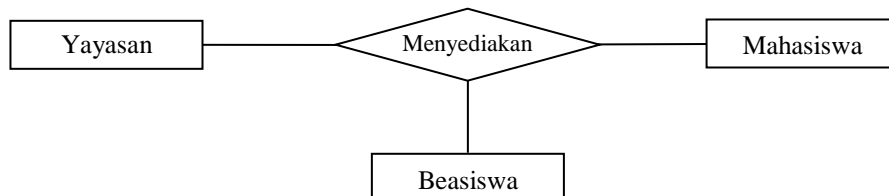
b. Binary relationship

Melibatkan 2 entitas / berderajat 2



c. Ternary Relationship

Melibatkan 3 entitas / berderajat 3



BAB III

TRANSFORMASI DIAGRAM E-R KE RELASI

Tujuan :

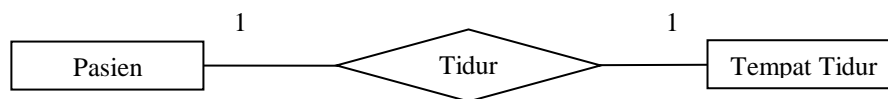
1. Memahami cara mentransformasi diagram ER ke relasi

3.1 Cardinality Ratio

1. Menjelaskan hubunga batasan jumlah keterhubungan satu entity dengan entity lainnya.
2. Menyatakan jumlah anggota entity yang terlibat dalam ikatan yang terjadi.
3. Tergantung pada aturan satu organisasi / aturan bisnis tertentu.
4. Jenis :
 - a. One to one, 1 : 1, satu ke satu
 - b. One to many, 1 : m, satu ke banyak
 - c. Many to many, m : n, banyak ke banyak

Contoh :

1 : 1



Seorang pasien hanya boleh menempati satu tempat tidur

Satu tempat tidur hanya boleh ditempati seorang pasien

Prabu	—	—	A11
Fadil	—	—	B12
Afif	—	—	A12

1 : m, m : 1



Seorang pegawai hanya boleh bekerja di satu departemen

Satu departemen bisa mempekerjakan banyak pegawai

Prabu	=====	Dep. Pertanian
Fadil	=====	Dep. Perindustrian
Afif	=====	Dep. Peternakan

m : n



Seorang mahasiswa boleh mengambil banyak mata kuliah

Satu mata kuliah boleh diambil banyak mahasiswa

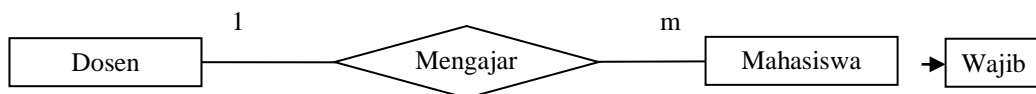
Prabu	=====	Kalkulus
Fadil	=====	Basis Data
Afif	=====	Matematika Diskrit

3.2 Partisipasi Hubungan

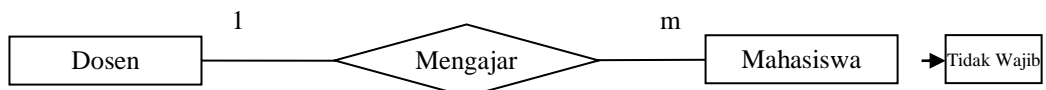
1. Partisipasi atau keterlibatan tiap anggota mentry dalam membentuk instan hubungan.
2. Dapat bersifat wajib (**Full / Total Participation**) atau tidak wajib (**Partial Participation**)
3. Penggambaran bisa menggunakan (berurutan lambang wajib / tidak wajib)

===== / — atau — / ----- atau — ☐ / — ☐

Misal :



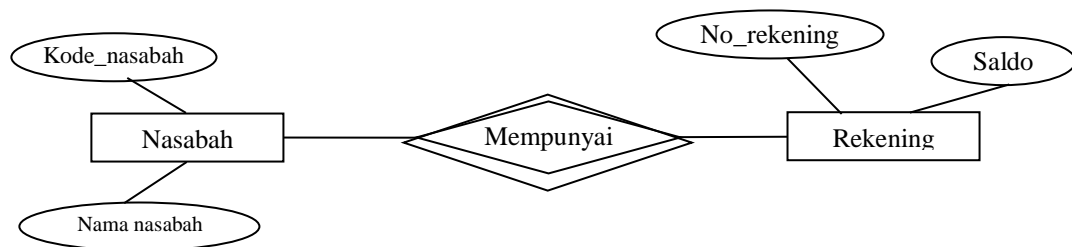
Setiap dosen harus membimbing minimal 1 orang mahasiswa.



Dosen bisa membimbing banyak mahasiswa atau tidak membimbing satu orang pun mahasiswa.

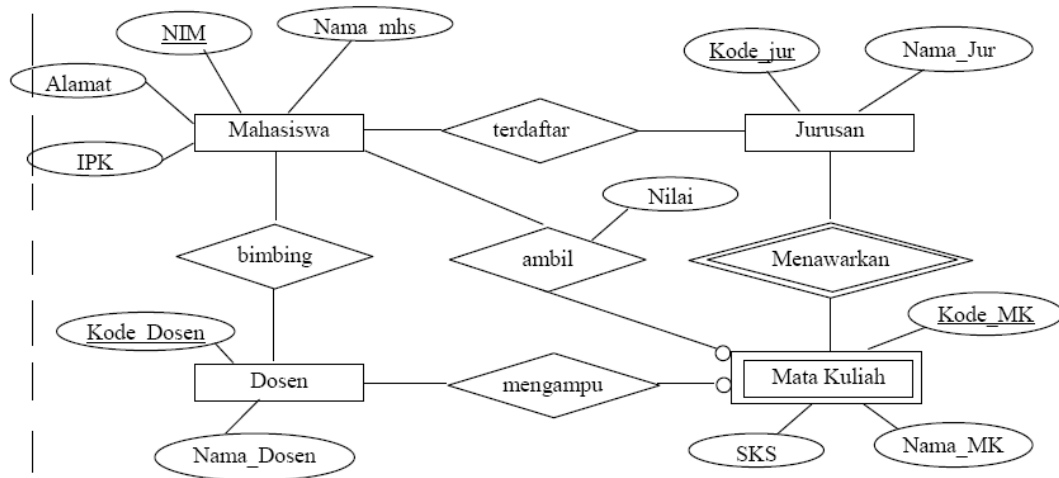
Entitas Kuat (Strong Entitas) dan Entitas Lemah (Weak Entity)

1. Entitas Kuat → keberadaannya tak tergantung keberadaan entitas lain.
2. Entitas Lemah → keberadaannya sangat tergantung keberadaan entitas lain.
3. Identifying owner
 - a. Entitas dimana entitas lemah bergantung.
 - b. Digambarkan dengan segiempat bertumpuk.
4. Identifying relationship
 - a. Relasi antara entitas lemah dan entitas kuat.
 - b. Digambarkan dengan diamond / wajik bertumpuk

**Soal :**

Sebuah Politeknik mempunyai 3 Buah Jurusan (TI, TM, TP). Setiap Mahasiswa hanya diperbolehkan mengambil 1 jurusan. Apabila mahasiswa jurusan TI akan mengambil Jurusan TM maka mahasiswa tersebut harus keluar dari jurusan TI dan pindah ke jurusan TM. Setiap mata kuliah hanya boleh diampu oleh seorang dosen, meskipun mata kuliah tersebut mata kuliah pararel (mata kuliah yang sama tetapi lebih dari 1 kelas). Apabila Dosen mempunyai kesibukan yang cukup padat seperti sedang menyelesaikan S2/S3 ataupun memegang jabatan struktural maka dosen tersebut diperkenankan untuk tidak mengajar mata kuliah apapun. Selain mengampu mata kuliah, setiap dosen wajib membimbing minimal 1 orang mahasiswa.

Jawaban :



BAB IV

MODEL DATA BERBASIS RECORD

4.1 Model Relasional

Model Data Relasional

1. Diperkenalkan pertama kali oleh E.F. Codd (1970)
2. Model data yang dominan dan mendasar pada produk andalan DBMS
3. Produk :
 - IBM DB2, Informix, Sybase, MsAcces, SQLServer, Foxpro, Paradox

Konsep Basis Data Relasional

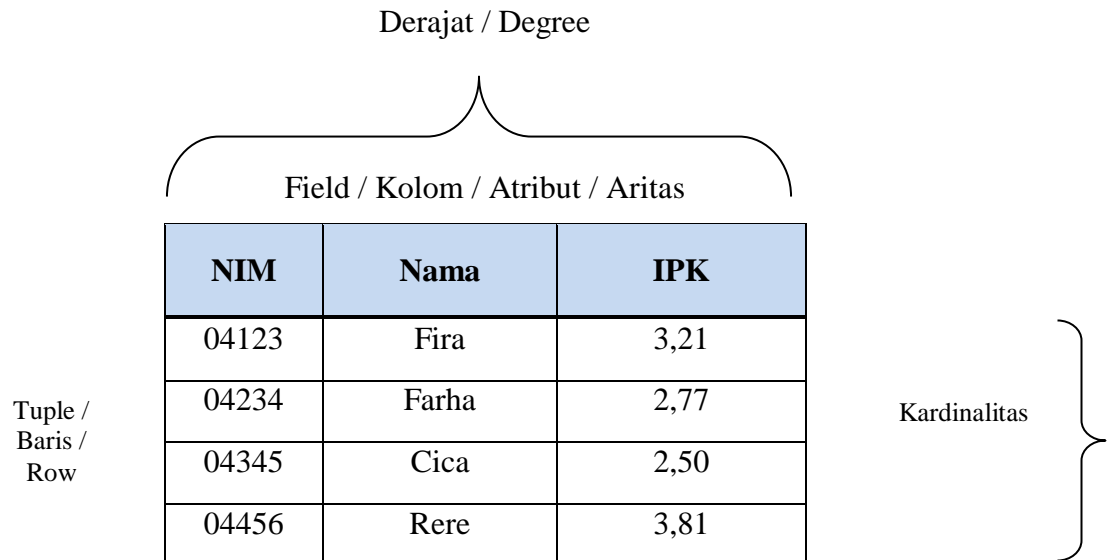
1. Basis data relasional merupakan kumpulan dari relasi dengan nama-nama relasi yang berbeda.
2. Berbasis pada teori himpunan dan matematika (aljabar dan kalkulus relational).
3. Relasi terbentuk dari 2 bagian :
 - a. Instan relasi : Tabel yang terdiri dari baris dan kolom
 - b. Skema : Menspesifikasikan nama dari suatu relasi ditambah nama dan type kolom

Istilah dalam Model Relasional

1. Relasi
 - a. Pada dasarnya berupa tabel dengan baris dan kolom yang bernama
2. Atribut / Field / Aritas / Kolom
 - a. Kolom yang bernama dalam suatu relasi
3. Degree → jumlah atribut yang dimiliki relasi
4. Tupel / Record / Baris / Row
 - a. Baris dari suatu relasi atau disebut juga elemen dari relasi
 - b. Berbeda dengan konsep record di file, tupel mempunyai jumlah kolom yang sama.
5. Kardinalitas → jumlah tupel dalam relasi, nilai kardinalitas adalah kondisi suatu saat dari suatu tabel jika tupel berubah maka kardinalitas ikut berubah.

6. Domain

- a. Himpunan nilai yang diijinkan pada suatu atribut

**Komponen Model Relasional**

1. Struktur Data

- a. Data-data diorganisasi dalam bentuk tabel dengan baris-baris dan kolom-kolom.

2. Manipulasi Data

- a. Operasi manipulasi dilakukan dengan menggunakan perintah SQL

3. Integritas Data

- a. Fasilitas untuk menspesifikasikan aturan bisnis yang memelihara integritas data saat dilakukan manipulasi.

Ketentuan / Properti Relasi

1. Setiap relasi dalam basis data harus mempunyai nama yang unik.
2. Setiap Atribut / Kolom mempunyai nama yang unik, urutan kolom tidak diperhatikan.
3. Relasi adalah himpunan tupel.
 - a. Setiap tupel bersifat unik / distinct [gambar relasi (1)] .
 - b. Karena berupa himpunan maka urutan keberadaan tupel tidak diperhatikan.

4. Setiap nilai pada perpotongan baris dan kolom tertentu bersifat atomic (bernilai tunggal), tidak diperkenankan atribut bernilai banyak [gambar relasi (2)].

NIM	Nama	Alamat	IPK
001	Prabu	Jalan Dukuh Jeruk 10	3,22
002	Fadil	Jalan Terusan 20	3,22
003	Afif	Jalan Macan 1 No. 10 A	2,7

Kode Mata Kuliah	Nama Mata Kuliah	Dosen
TIKK1032	Rangkaian Listrik	Muh. Lukman Sifa
TIKB2063	Pemrog. Berorientasi Objek	Willy Permana Putra
TIKB2093	Aplikasi Mikroprosesor	A.Sumarudin
TIKB1014	Arsitektur Komputer	Munengsih SB.

Gambar Relasi (1)

NIM	Nama	Mata Kuliah
001	Prabu	Kalkulus, Matematika Diskrit
002	Fadil	Basis Data, Kalkulus
003	Afif	Basis Data, Kalkulus, Pemrograman Web

NIM	Nama	Mata Kuliah
001	Prabu	Kalkulus, Matematika Diskrit
002	Fadil	Basis Data, Kalkulus
003	Afif	Basis Data, Kalkulus, Pemrograman Web

NIM	Nama	Mata Kuliah
001	Prabu	Kalkulus
001	Prabu	Matematika Diskrit
002	Fadil	Basis Data
002	Fadil	Kalkulus
003	Afif	Basis Data
003	Afif	Kalkulus
003	Afif	Pemrograman Web

Gambar Relasi (2)

Skema Relasi

1. Skema relasi terdiri dari :
 - a. Nama Relasi → Harus unik
 - b. Nama atribut relasi → Harus unik dan diasosiasikan dengan nama domain

Contoh :

Mahasiswa (nim : char (10), nama : char (40), alamat : char (40), ipk : decimal (4,2))

- a. Pengembangan berikutnya adalah disertakannya konstrain integritas
2. Instan Relasi [gambar relasi (1)] .

Primary dan Foreign Key

1. *Primary Key* (Kunci Primer / Utama)
 - a. Atribut atau kombinasi atribut yang secara unik mengidentifikasi setiap baris dalam relasi.
2. *Foreign Key* (Kunci Tamu)
 - a. Atribut atau kombinasi pada relasi yang berfungsi sebagai kunci primer pada relasi lain pada basis data yang sama.
 - b. Digunakan untuk melakukan referensi ke tupel pada relasi lain (relasi yang menjadikan kunci tersebut sebagai kunci primer).

Contoh Relasi

NIM	Nama_mhs	Alamat	IPK
001	Prabu	Jalan Dukuh Jeruk 10	3,22
002	Fadil	Jalan Terusan 20	3,22
003	Afif	Jl. Macan 1 No. 10 A	2,7

Gambar : Tabel Mahasiswa

Kode_mk	Nama_mk	SKS
A01	Matematika Diskrit	2
A02	Rangkaian Listrik	2
A03	Pemrog. Berorientasi Objek	3

Gambar : Tabel Mata Kuliah

NIM	Kode_mk
001	A01
001	A03
002	A01
003	A02
003	A03

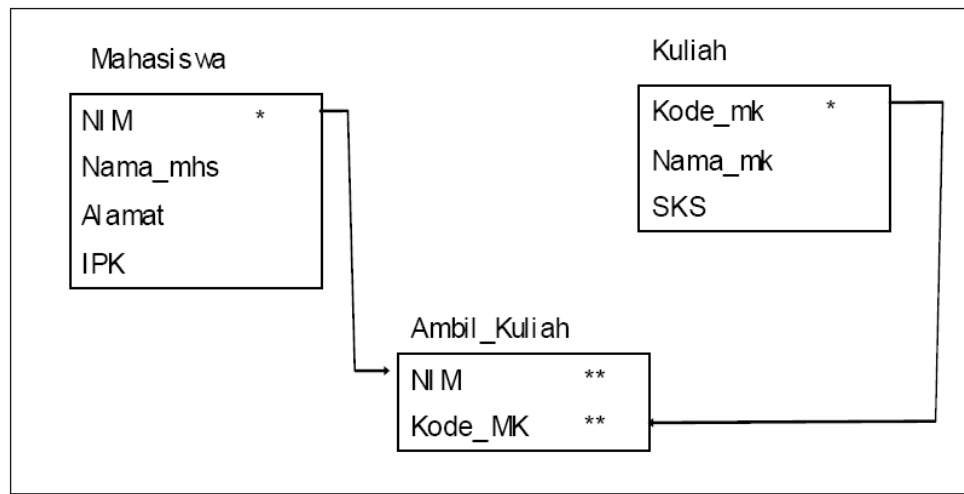
Gambar : Tabel Ambil_mata_kuliah

Skema Relasi :

Mahasiswa (nim : char (10), nama_mhs : char (25), alamat : char (40), ipk decimal (4,2))

Kuliah (kode_mk : char (8), nama_mk : char (30), sks : integer))

Ambil_kuliah (nim : char (10), kode_mk : char (8))



BAB V

NORMALISASI

Tujuan :

1. Mengetahui definisi normalisasi
2. Mengetahui tujuan dari normalisasi
3. Mengetahui tahapan-tahapan normalisasi
4. Mampu melakukan proses normalisasi dari sebuah basis data

5.1 Permasalahan Redundansi

1. Redundansi : Perulangan yang berlebihan
2. Redundansi adalah alasan utama dilakukannya normalisasi
3. Redundansi menyebabkan :
 - a. Pemborosan ruang penyimpanan
 - b. Anomali pada saat insert (simpan), Update (edit / pembaharuan), Delete (hapus).
 - c. Bisa menyebabkan inkonsistensi

Contoh Redundansi

NIM	Nama	Alamat	Hobby
001	Prabu	Jalan Dukuh Jeruk 10	Membaca
001	Prabu	Jalan Dukuh Jeruk 10	Menari
002	Afif	Jl. Macan 1 No. 10 A	Membaca
002	Afif	Jl. Macan 1 No. 10 A	Menyanyi
002	Afif	Jl. Macan 1 No. 10 A	Nonton
003	Fadil	Jl. Terusan No. 10	Berburu
003	Fadil	Jl. Terusan No. 10	Berenang
004	Fairus	Jl. Kembang No. 12	Bersepeda

Kunci Utama ? → NIM + Hobby
 → Tidak Boleh Kosong / Null

Anomali Yang Terjadi

1. *Anomali Insert* (Penyisipan)
 - a. Keadaan yang tidak diinginkan atau munculnya masalah saat akan menyisipkan data baru ke dalam relasi dengan struktur yang tidak lengkap.
 - b. Data baru bernama Fikri tetapi hobby belum tercatat ... ?
2. *Anomali Update* (Pembaharuan)
 - a. Keadaan di mana apabila satu nilai atribut perlu diperbaharui, jika lebih dari pada satu data yang terlibat yang disebabkan adanya pengulangan data maka apabila tidak semua data diperbaharui akan menimbulkan inkonsistensi data
 - b. Afif berpindah alamat ... ?
3. *Anomali Delete* (Penghapusan)
 - a. Keadaan dimana apabila satu data dihapuskan pada suatu relasi, terjadi kehilangan data lain yang masih diperlukan.
 - b. Data hobby bersepeda dihilangkan → Data Fairuz ikut hilang.
4. Penyebab Redundansi
 - a. Perulangan fakta yang sama
 - b. Adanya atribut turunan
 - Umur (diperoleh dari tanggal lahir)
 - Subtotal (diperoleh dari Qty x Harga)
5. Solusi Redundansi
 - a. Penghilangan data turunan
 - b. Dekomposisi
 - a. Lossy decomposition / lossy-join decomposition (kehilangan info)
 - b. Lossless – join decomposition (tidak kehilangan info)
6. Redundansi > < Duplikasi
 - a. Duplikasi bisa diterima karena menggambarkan fakta yang berbeda

5.2 Kebergantungan

1. Kebergantungan fungsional (functional dependency / FD)
 - a. Diberikan sebuah relasi R, atribut Y dari R adalah bergantung fungsi pada atribut X dari R jika dan hanya jika setiap nilai X dalam R punya hubungan dengan tepat satu nilai Y dalam R.
 - b. Digambarkan : $X \rightarrow Y$
 - c. Sisi kiri (X) disebut sebagai determinan atau penentu
 - d. Untuk setiap nilai X tertentu pasti akan didapatkan nilai Y yang sama

Contoh Kebergantungan

1. $IDPegawai \rightarrow NamaPegawai$
2. $NIM, Kode_MK \rightarrow Nilai_MK$
 - a. Kebergantungan pada lebih dari satu atribut
3. Kebergantungan Fungsional Penuh :
 - a. $IDPegawai \rightarrow Tgl_lahir \dots\dots(1)$
 - b. $IDPegawai, NamaPegawai \rightarrow Tgl_lahir \dots\dots(2)$
 - c. No 2 : Namapegawai sebenarnya tidak diperlukan untuk memperoleh Tgl_lahir
 - d. No 1 : Kebergantungan Fungsional Penuh
4. Kebergantungan Transitif
 - a. Kebergantungan pada suatu atribut bukan kunci utama

5.3 Proses Normalisasi

1. Penormalan adalah proses menguraikan relasi-relasi yang bermasalah mengikuti aturan atau ciri-ciri tertentu.
2. Penormalan dilaksanakan langkah demi langkah, yaitu dari satu bentuk normal ke satu bentuk normal yang lebih tinggi.
3. Setiap bentuk normal mempunyai syarat-syarat kelayakan tertentu yang mesti dipenuhi sebelum relasi tersebut diuji untuk bentuk normal yang lebih tinggi.
4. Ada 6 tingkat bentuk normal
 - a. Bentuk Normal Pertama (1NF)
 - b. Bentuk Normal Kedua (2NF)
 - c. Bentuk Normal Ketiga (3NF)

- d. Bentuk Normal Boyce Codd (BCNF)
 - e. Bentuk Normal Keempat (4NF) – Fagin
 - f. Bentuk Normal Kelima (5NF) – Fagin
5. Untuk relasi sederhana biasanya sudah dalam kondisi yang baik pada 2 NF atau 3 NF.

5.4 Tahapan Normalisasi

1. Kondisi Relati tidak normal (unnormalized) adalah kondisi data apa adanya, tidak mengikuti suatu format tertentu termasuk ketidaklengkapan ataupun redundansi.
2. Contoh :

NIM	Nama	Mata Kuliah
001	Prabu	Kalkulus, Matematika Diskrit
002	Fadil	Basis Data, Kalkulus
003	Afif	Basis Data, Kalkulus, Pemrograman Web

NIM	Nama	Mata Kuliah
001	Prabu	Kalkulus
		Matematika Diskrit
002	Fadil	Basis Data
		Kalkulus
003	Afif	Basis Data
		Kalkulus
		Pemrograman Web

Contoh Tahapan dalam Normalisasi

Bentuk UnNormalized (Tidak Normal)

Berikut ini merupakan contoh nyata dari suatu faktur pembelian saat dilakukan penulisan ulang ke dalam dokumen pembelian.

No. Nota	Tanggal	Kode Sup	Nama Sup	Kode Brg	Nama Brg	Qty	Harga	Jumlah	Total
001	01/01/05	U01	Unilever	L001	Sabun Lux	2	500	1000	1000
002	02/01/05	I01	Indofood	M001	Indomie	5	700	3500	9000
				S001	Supermie	10	550	5500	
003	04/01/05	I01	Indofood	M001	Indomie	10	700	7000	7000

1. Terdapat Record yang tidak lengkap
2. Terlihat beberapa Duplikasi yang tidak perlu (redundansi)

Bentuk Normal Pertama (1NF)

1. Suatu relasi memenuhi bentuk normal pertama (1NF) jika dan hanya jika setiap Atributnya bersifat atomik

No. Nota	Tanggal	Kode Sup	Nama Sup	Kode Brg	Nama Brg	Qty	Harga	Jumlah	Total
001	01/01/05	U01	Unilever	L001	Sabun Lux	2	500	1000	1000
002	02/01/05	I01	Indofood	M001	Indomie	5	700	3500	9000
002	02/01/05	I01	Indofood	S001	Supermie	10	550	5500	9000
003	04/01/05	I01	Indofood	M001	Indomie	10	700	7000	7000

Bentuk Normal Kedua (2NF)

1. Telah memenuhi 1NF
2. Atribut bukan kunci harus bergantung fungsional pada primary key (kunci utama).
3. Primary Key harus unik dan bisa menjadi identitas atribut lain yang menjadi anggotanya.
4. Perlu dilakukan dekomposisi dan penghapusan atribut turunan.

Kode Sup	Nama Sup
U01	Unilever
I01	Indofood

Tabel : Suplier

Kode Brg	Nama Brg
L001	Sabun Lux
S001	Supermie
M001	Indomie

Tabel : Barang

No. Nota	Tanggal	Kode Sup	Kode Brg	Qty	Harga	Total
001	01/01/05	U01	L001	2	500	1000
002	02/01/05	I01	M001	5	700	9000
002	02/01/05	I01	S001	10	550	9000
003	04/01/05	I01	M001	10	700	7000

Tabel : Nota

Bentuk Normal Ketiga (3NF)

1. Telah memenuhi 2NF
2. Semua atribut bukan primer tidak punya hubungan yang transitif
3. Semua atribut bukan kunci hanya bergantung pada primary key
4. Untuk contoh data pembelian, pada relasi nota terdapat permasalahan :
 - a. Atribut QTY dan Harga tidak tergantung penuh pada nomor nota tetapi juga bergantung fungsional terhadap kode barang (ketergantungan transitif).
 - b. Masih terdapat redundansi (nonota, tanggal, kodesup, total) jika pada satu nota dilakukan pembelian lebih dari 1 item barang.

Kode Sup	Nama Sup
I01	Unilever
I01	Indofood

Tabel : Suplier

Kode Brg.	Nama Brg
L001	Sabun Lux
S001	Supermie
M001	Indomie

Tabel : Barang

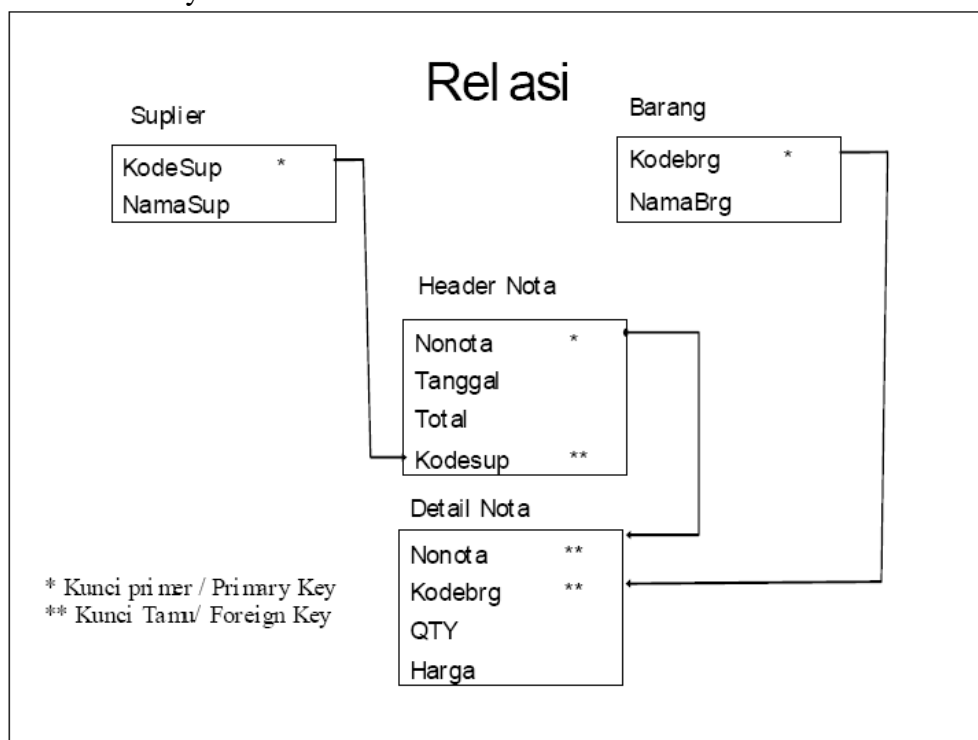
No. Nota	Tanggal	Kode Sup	Total
001	01/01/05	U01	1000
002	02/01/05	I01	9000
003	04/01/05	I01	7000

Tabel : Header_Nota

No. Nota	Kode Brg.	Qty	Harga
001	L001	2	500
002	M001	5	700
002	S001	10	550
003	M001	10	700

Tabel : Detail_Nota

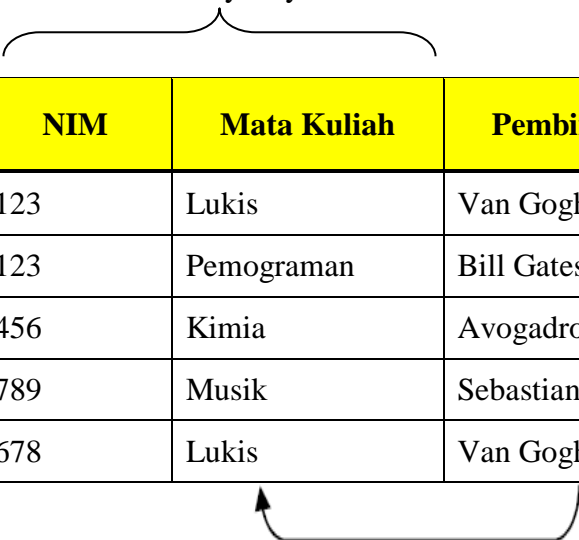
Hasil Relasinya :



BCNF

1. Meskipun sudah memenuhi 3NF namun masih memiliki lebih dari 1 kunci kandidat.

Primary Key



NIM	Mata Kuliah	Pembimbing	IP
123	Lukis	Van Gogh	3,9
123	Pemograman	Bill Gates	3,3
456	Kimia	Avogadro	3,2
789	Musik	Sebastian Bach	3,7
678	Lukis	Van Gogh	3,5

Permasalahan

1. Kebergantungan transitif = kebergantungan antara 2 atribut bukan kunci → sudah tidak ditemukan dan sudah memenuhi 3 NF
2. Mata kuliah bergantung secara fungsional terhadap pembimbing
3. Update anomaly
 - a. Bila untuk mata kuliah Lukis, pembimbing Van Gogh diganti oleh Affandi maka harus dilakukan 2 (atau lebih) perubahan.
4. Insertion Anomaly
 - a. Bila akan menambahkan nama Soeharto untuk membimbing mata kuliah Ilmu Politik tidak dapat dilakukan hingga paling sedikit 1 mahasiswa mengambil mata kuliah tsb.
5. Delection Anomaly
 - a. Jika mahasiswa dengan NIM = 789 dihapus maka data Sebastian Bach juga akan hilang.

Penyelesaian dari permasalahan tersebut di atas

NIM	Pembimbing	IP
123	Van Gogh	3,9
123	Bill Gates	3,3
456	Avogadro	3,2
789	Sebastian Bach	3,7
678	Van Gogh	3,5

Pembimbing	Mata Kuliah
Van Gogh	Lukis
Bill Gates	Pemrograman
Avogadro	Kimia
Sebastian Bach	Musik

Efek Negatif Normalisasi

1. Performance Problem
 - a. Masalah performance yang terutama adalah saat akan menampilkan data, apabila sebelumnya hanya perlu membuka satu tabel maka setelah dilakukan normalisasi setiap menampilkan data diperlukan dua atau lebih tabel yang harus dibuka.
2. Referential Integrity Problem
 - a. Adanya permasalahan Parent – Child problem saat penambahan data baru (kode belum ada di parent) dan penghapusan (kode masih ada di child)
 - b. Bisa diatasi dengan referential integrity yang terdapat dalam DBMS, tetapi ada kemungkinan mempunyai resiko tinggi.
3. Permasalahan Pemrograman
 - a. Kesulitan dalam proses pemrograman
 - b. Kesulitan dalam pemberian fasilitas kemudahan user friendly, misalnya mengurutkan data dengan index.

Denormalisasi

1. Proses normalisasi basis data yang ketat (tanpa redundansi dan tanpa anomali) meningkatkan efisiensi dan konsistensi data.
2. Terkadang berakibat tidak efisiensinya proses dan kompleksitas dalam pemrograman.
3. Kinerja Sistem turun
4. Denormalisasi : Proses transformasi relasi yang telah normal menjadi relasi yang tidak normal.

Alasan Denormalisasi

1. Beberapa hal penting dalam pengembangan sistem :
 - a. Konsistensi dan integritas data
 - b. Kecepatan Proses
 - c. Keamanan data
 - d. Kemudahan Pembuatan Program
2. Saat melakukan denormalisasi biasanya dikarenakan 3 hal terakhir
3. Denormalisasi harus dilakukan hati-hati agar tidak kehilangan hal yang pertama di atas.

Akibat Denormalisasi

1. Kemungkinan terjadinya inkonsistensi maupun tidak terpenuhinya integritas data
2. Efisiensi penyimpanan berkurang
3. Denormalisasi pada suatu data bisa berakibat menurunkannya kinerja pengolahan data yang lain
4. Bila terjadi perubahan aturan bisnis maka bisa mengakibatkan kontra produktif.

Teknik Denormalisasi

1. Bila ada relasi 1 ke 1 maka ada kemungkinan kedua relasi tersebut digabungkan.
 - a. Mis :
 - Mhs (nim c(10); nama c(30); alamat c(50))
 - Ortu (idOrtu c(5); nim c (10); nama_ortu c(30); alamat_ortu c(50))
 - b. Hasil denormalisasi
 - Mhs (nim c(10); nama c(30); alamat c(50); nama_ortu c(30); alamat_ortu c(50))
2. Data Acuan → apabila suatu data acuan jarang sekali berubah dan sering dipakai maka data acuan bisa dimasukkan ke dalam relasi yang menggunakannya.
 - a. Misal :
 - Barang (kode c(8); nama_brg c(30); kode _sat c(2))
 - Satuan)kode_sat c(2); nama_sat c(15))
 - b. Menjadi
 - Barang (Kode c(8); nama_brg c(30); kode_sat c(2); nama_sat c(15))
 - Satuan (Kode_sat c(2); nama_sat c(15))
3. Atribut Turunan → Terkadang apabila suatu atribut turunan sering diakses maka sebaiknya atribut tersebut dimunculkan dalam relasi
 - a. Misal :
 - Detail_beli (id_beli c(10); kode_brg c(8); qty n(5); harga n(7); disc1 n(6,2); disc2 n(6,2); disc3 n(6,2); rpDisc1 n(7); rpDisc2 n(7); rpDisc3 n(7))
4. Atribut Terkodekan → Apabila dalam suatu atribut sudah mengandung suatu arti tertentu namun ada kemungkinan bahwa arti tersebut bisa berubah pada suatu saat maka arti kode tersebut bisa dibuat atribut baru
 - a. Misal :
 - Barang (kode_brg c(5); nama_brg c(30))
 - Contoh Kode_brg : 01001, 02001 dll
 - 2 digit awal berarti kode kelompok
 - Kelompok (kode_klmpk c(2); nama_klmpk c(15))

- b. Menjadi :
 - Barang (kode_brg c(5); nama_brg c(30); kode_klmpk c(2))
 - Kelompok (kode_klmpk c(2); nama_klmpk c(15))
- 5. Data Rekapitulasi → untuk mempercepat Pelaporan terkadang dibutuhkan suatu data rekap dari data transaksi.
 - a. Misal :
 - JualBahan (bulan c(2); kode_brg c(5); qty_jual n(7); jumlah_jual n(8))
 - Atribut jumlah_jual sebenarnya merupakan hasil perhitungan $qty * \text{harga}$ per barang setiap ada transaksi penjualan
 - Qty_jual merupakan perhitungan total qty per barang setiap transaksi penjualan
- 6. Tabel Laporan → Tabel Laporan biasanya bersifat temporary table / tabel sementara.
 - a. Digunakan untuk memudahkan dalam pembuatan laporan
 - b. Tidak memperhitungkan terjadinya redundansi
 - c. Misal :
 - Lap_jua (tgl d; nonota c(10); kode_brg c(5); nama_brg c(30)); qty n(3); harga n(7); jumlah n(8)).

BAB VI
DASAR-DASAR
SQL (Structured Query Language)

Tujuan :

1. Mengetahui definisi SQL
2. Mengetahui definisi DDL
3. Mengetahui fungsi sintaks DDL
4. Mampu menggunakan sintaks DDL

6.1 Definisi SQL

SQL adalah bahasa standar dalam basis data yang digunakan untuk melakukan manipulasi data. Standardisasi bahasa ini dilakukan oleh ANSI tahun 96, 89, 92 dan 99, dimana tiap perubahan tahun dilakukan peningkatan kemampuan SQL. Pada perkembangan saat ini standar yang paling banyak digunakan adalah standar ANSI 92. Hampir semua DBMS menggunakan SQL sebagai fasilitas untuk memanipulasi data seperti Oracle, SQLServer, MySQL, PostgreSQL, Foxpro dsb.

Meskipun awalnya hanya merupakan bahasa untuk memanipulasi data, pada perkembangannya SQL juga dapat digunakan untuk melakukan definisi data maupun control (*security*) terhadap data. Sehingga bahasa Query ini dibagi menjadi 3 bagian :

1. DDL (Data Definition Language)
2. DML (Data Definition Language)
3. DCL (Data Control Language)

DDL (Data Definition Language)

DDL merupakan bahasa yang digunakan untuk membuat atau memodifikasi database dan tabel, Perintah DDL a.l :

CREATE → Untuk membuat

Syntak :

1. Membuat database

```
CREATE DATABASE nama_database
```

2. Membuat tabel

```
CREATE TABLE nama_tabel (field1 type_data1 (lebar_data1), field2 type_data2 (lebar_data2))
```

Apabila akan menambahkan konstrain integritas PRIMARY KEY maka syntaknya adalah sbb :

```
CREATE TABLE nama_tabel (field1 type_data1 (lebar_data1) PRIMARY KEY, field2 type_data2 (lebar_data2))
```

Catatan : yang akan dijadikan primary key adalah field1

Misal :

```
CREATE TABLE mahasiswa (nim char(10), nama char(30), jurusan char (2), ipk float(4,2))
```

DROP → untuk Menghapus :

Syntak :

1. Menghapus database

```
DROP DATABASE nama_table
```

2. Menghapus tabel

```
DROP TABLE nama_table
```

ALTER → untuk Memodifikasi Tabel

Syntak :

1. Menambah field baru

```
ALTER TABLE nama_tabel ADD COLUMN field_baru type_data (lebar_data)
```

Misal : ALTER TABLE mahasiswa ADD COLUMN alamat varchar(80)

2. Menghapus field

```
ALTER TABLE nama_tabel DROP COLUMN field_yang_dihapus
```

Misal : ALTER TABLE mahasiswa DROP COLUMN alamat

3. Mengedit / mengganti field

```
ALTER TABLE nama_tabel CHANGE [COLUMN] field_nama field_baru
type_data (lebar_data)
```

Misal : ALTER TABLE mahasiswa CHANGE nama nama_mhs char(40)

DML (Data Manipulation Language)

DML merupakan bahasa untuk memanipulasi data (membaca/menampilkan, menambah, mengedit, menghapus)

Perintah DML a.l :

1. Untuk membaca atau menampilkan data

```
SELECT daftar_field_yang_akan_ditampilkan
FROM nama_tabel
[WHERE kriteria_data_yang_akan_ditampilkan]
```

6.2 Perintah Dasar SQL

1. Menampilkan Database

Adakalanya didalam sebuah database server sistem yang besar terdapat beberapa atau banyak database. Untuk itu kita membutuhkan informasi database apa saja yang ada pada database server sistem tersebut. Sintaks untuk menampilkan database adalah :

```
SHOW DATABASES;
```

Penjelasan : perintah SHOW DATABASES akan menampilkan nama-nama database yang ada pada storage database server tersebut.

2. Memilih database yang akan digunakan

Tabel-tabel yang kita gunakan dalam pemrograman basis data bisa saja berlokasi pada database yang berbeda. Untuk dapat melakukan manipulasi tabel-tabel tersebut sebelumnya harus ditunjuk dulu tabel tersebut berada pada database mana. Sintaks/perintahnya adalah :

```
USE [nama_database] ;
```

3. Menampilkan tabel

Untuk mengetahui tabel-tabel apa saja yang ada didalam database, sintaksnya adalah :

```
SHOW TABLES;
```


BAB VII

OPERASI DASAR SQL

Tujuan :

1. Mengetahui sintaks-sintaks dan operasi dasar SQL
2. Mengetahui fungsi perintah SQL
3. Mampu menggunakan perintah SQL

7.1 Pengantar Operasi Relasi

Struktur dasar dari ekspresi dalam SQL mengandung 3 klausa utama : ***SELECT***, ***FROM***, dan ***WHERE***.

1. Klausa ***SELECT***
 - Klausa ***select*** digunakan untuk mendaftarkan atribut-atribut yang dikehendaki sebagai hasil dari suatu query, atau bisa dikatakan bahwa klausa ***select*** digunakan untuk menampilkan field-field dari suatu tabel.
2. Klausa ***FROM***
 - Klausa ***from*** digunakan untuk mendaftarkan relasi-relasi yang digunakan pada proses pencarian atau secara sederhananya, klausa ***from*** digunakan untuk menentukan atribut-atribut akan diambil dari tabel mana.
3. Klausa ***WHERE***
 - Klausa ***where*** digunakan untuk mendaftarkan kriteria-kriteria pencarian, atau dengan kata lain, klausa ***where*** digunakan untuk menyeleksi data yang akan ditampilkan sesuai dengan kriteria.

7.2 Modifikasi Basis Data

Beberapa operator relasi yang digunakan pada saat akan dibutuhkan suatu kriteria tertentu untuk menampilkan data adalah :

Operator	Arti
=	Sama dengan
>	Lebih besar
> =	Lebih besar atau sama dengan
<	Lebih kecil
< =	Lebih kecil atau sama dengan
<>	Tidak sama dengan
LIKE	Mengandung suatu kata/huruf tertentu
BETWEEN	Rentang antara dua nilai

1. Sedangkan operator logika yang sering digunakan adalah AND, OR dan NOT.

Misal :

- a. Menampilkan nim dan nama semua mahasiswa

```
SELECT nim, nama FROM mahasiswa
```

- b. Menampilkan nim dan nama mahasiswa jurusan TI

```
SELECT nim, nama FROM mahasiswa WHERE jurusan = "TI"
```

- c. Menampilkan semua field dari tabel mahasiswa jurusan TI

```
SELECT * FROM mahasiswa WHERE jurusan = "TI"
```

- d. Menampilkan nama mahasiswa yang berawalan "PAR"

```
SELECT nama FROM mahasiswa WHERE nama LIKE "PAR%"
```

- e. Menampilkan nim, nama dan IPK mahasiswa yang mempunyai IPK 2,5 sampai 3,2

```
SELECT nim, nama, ipk FROM mahasiswa WHERE ipk BETWEEN 2.5 AND 3.2
```

- f. Menampilkan nim, nama dan alamat mahasiswa jurusan TI yang berjenis kelamin wanita

```
SELECT nim, nama, alamat FROM mahasiswa WHERE jurusan = "TI" AND jenis_kel = "WANITA"
```

2. Untuk menambah data baru

Syntax :

```
INSERT INTO nama_tabel (field1, field2, ...) VALUES (values1, values2,...)
```

Field dan value harus berjumlah sama dan masing-masing berpasangan, artinya : value1 akan diisikan ke field1, value2 akan diisikan ke field2, dst.

Misal :

```
INSERT INTO mahasiswa (nim, nama) VALUES ("05023562","PRABU")
```

3. Untuk mengedit data

Syntax :

```
UPDATE nama_tabel SET field1 = value1, field2 = value2, ...
```

```
[WHERE kriteria]
```

Misal : Untuk mengganti nama mahasiswa menjadi AFIF untuk nim 05023562 UPDATE mahasiswa SET nama = "AFIF" WHERE nim = "05023562".

4. Untuk menghapus data

Syntax :

```
DELETE FROM nama_tabel
```

```
[WHERE Kriteria]
```

Misal : Untuk menghapus data mahasiswa yang mempunyai nim 050235562 DELETE FROM mahasiswa WHERE nim = "05023562".

DCL (Data Control Language)

DCL merupakan bahasa yang digunakan untuk pengaturan akses seorang user terhadap data.

Catatan : user sudah harus terdaftar

1. Memberikan hak akses kepada user

```
Syntax : GRANT hak_atau_privileges  
          ON Nama_Database>Nama_Tabel  
          TO Nama_User
```

Misal :

(contoh berikut menggunakan MySQL)

Seorang user (prabu dengan host : localhost) diperkenankan untuk membaca dan menambah data mahasiswa dalam database kampus tetapi tidak diperkenankan untuk melakukan hal yang lain seperti mengedit, menghapus, dll.

```
GRANT select, insert ON kampus.mahasiswa TO prabu@localhost
```

2. Mencabut hak akses dari user

Syntak :

```
REVOKE hak_atau_privileges  
ON Nama_Database>Nama_Tabel  
FROM Nama_User
```

Misal :

(contoh berikut menggunakan MySQL)

Hak yang diberikan kepada prabu di atas dicabut

```
REVOKE select, insert ON kampus.mahasiswa FROM prabu@localhost
```