2018

Modul
Sistem Basis Data

Sekolah Tinggi Manajemen Informatika dan Komputer
Nusa Mandiri

Kata Pengantar

Puji syukur penulis panjatkan kehadirat Allah SWT, yang telah memberikan rahmat dan hidayahnya sehingga modul Sistem Basis Data ini dapat terselesaikan dengan baik. Selanjutnya modul ini disusun untuk memberikan gambaran bagi mahasiswa yang mempelajari Sistem Basis Data. Dengan menggunakan metode ***"Belajar Cepat Dan Menyenangkan"*** karena modul ini disertai contoh kasus, sehingga lebih memudahkan anda dalam memahami Sistem Basis Data.

Tak lupa penulis mengucapkan banyak terima kasih kepada semua pihak yang telah membantu dengan tenaga dan pikirannya, terima kasih juga kepada rekan–rekan instruktur, dosen dan semuanya yang tidak bisa disebutkan satu persatu, yang selalu mendukung penulis sehingga modul ini sehingga dapat selesai sesuai yang kita inginkan semua.

Penulis menyadari masih banyak kekurangan dalam penyusunan modul ini. Untuk itu saran dan kritik yang membangun sangat penulis harapkan guna perbaikan dan pengembangan modul ini kedepan.

Akhir kata penulis berharap semoga modul Sistem Basis Data ini dapat dipergunakan sebaik-baiknya dan dapat dijadikan referensi untuk mahasiswa umum yang ingin mempelajari Sistem Basis Data.

Jakarta,

Penulis

Daftar Isi

BAB I
Konsep Dasar Basis Bata (Database)

1.1 Pengenalan Basis Data

Basis data adalah kumpulan data yang saling berhubungan secara logis dan didesain untuk mendapatkan data yang dibutuhkan oleh suatu organisasi (Indrajani, 2015).

Basis data sudah banyak digunakan dalam berbagai jenis aplikasi, mulai dari aplikasi sederhana, seperti aplikasi pengelolaan nomor telepon sampai dengan aplikasi kompleks, seperti aplikasi pembayaran gaji karyawan perusahaan.

Konsep Dasar Basis Data

BASIS DATA adalah suatu susunan/kumpulan data operasional lengkap dari suatu organisasi/perusahaan yang diorganisir/dikelola dan simpan secara terintegrasi dengan menggunakan metode tertentu dengan menggunakan komputer sehingga mampu menyediakan informasi yang diperlukan pemakainya.

SISTEM BASIS DATA adalah kumpulan dari program aplikasi yang berinteraksi dengan basis data bersama dengan *Database Management System* (*DBMS*) dan basis data itu sendiri (Connolly dan Begg, 2010).

1.2 Komponen Sistem Basis Data

Terdapat 4 komponen pokok dari sistem basis data:

A. DATA, dengan ciri-ciri :

1. Data disimpan secara terintegrasi (*Integrated*)
   Terintegrated yaitu Database merupakan kumpulan dari berbagai macam file dari aplikasi-aplikasi yang berbeda yang disusun dengan cara menghilangkan bagian-bagian yang rangkap (*redundant*)
2. Data dapat dipakai secara bersama-sama(*shared*)
   Shared yaitu Masing-masing bagian dari database dapat diakses oleh pemakai dalam waktu yang bersamaan, untuk aplikasi yang berbeda.

Ada 3 jenis data pada sistem basis data, yaitu:

1. Data operasional dari suatu organisasi, berupa data yang disimpan didalam database
2. Data masukan (input data), data dari luar sistem yang dimasukan melalui peralatan input (keyboard) yang dapat merubah data operasional
3. Data keluaran (output data), berupa laporan melalui peralatan output sebagai hasil dari dalam sistem yang mengakses data operasional

B. Perangkat Keras (HARDWARE)

Terdiri dari semua peralatan perangkat keras komputer yang digunakan untuk pengelolaan sistem database.

Perangkat keras yang terdapat dalam sebuah sistem basis data adalah:

1. Komputer (satu untuk sistem *stand-alone* atau lebih dari satu untuk sistem jaringan)
2. Memori sekunder *on-line* (*Harddisk*)
3. Memori sekunder *off-line* (*Tape atau Removeble Disk*) untuk *backup* data
4. Media/perangkat komunikasi (untuk sistem jaringan)

C. Perangkat Lunak (SOFTWARE)

Berfungsi sebagai perantara (*interface*) antara pemakai dengan data phisik pada database, dapat berupa :

1. Database Management System (DBMS)
2. Program-program aplikasi & prosedur-prosedur

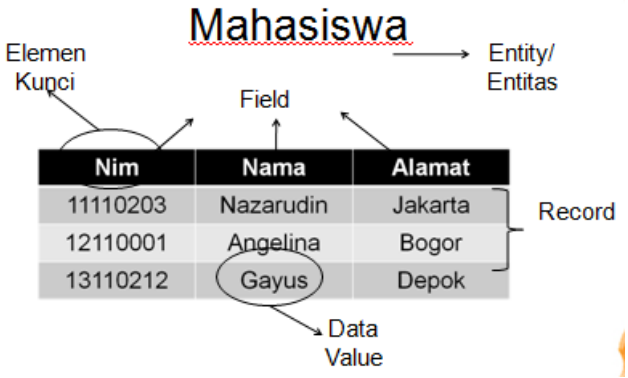
D. Pemakai (USER)

Adalah pengguna basis data yang berinteraksi secara tidak langsung dengan basis data melalui program aplikasi basis data dan DBMS**.** Terbagi menjadi 3 klasifikasi :

1. Database Administrator (DBA), yang membuat basis data dan mengontrol akses ke basis data.
2. Programmer, yang membuat aplikasi basis data yang digunakan oleh DBA dan pemakai akhir.
3. Pemakai akhir (End user) yang melakukan penambahan, penghapusan, pengubahan, dan pengaksesan data.

Istilah-istilah Dalam Sistem Basis Data

a. Enterprise yaitu suatu bentuk organisasi
   Contoh Enterprise: Sekolah, Rumah Sakit
   Sekolah : Database Nilai
   Rumah sakit : AdministrasiPasien
b. Entitas yaitu suatu obyek yang dapat dibedakan dengan objek lainnya
   Contoh :
   Database Nilai → entitas: Mahasiswa, Matapelajaran
   Database AdministrasiPasien → entitas: Pasien, Dokter, Obat
c. Atribute/field yaitu setiap entitas mempunyai atribut atau suatu sebutan untuk mewakili suatu entitas.
   Contoh :
   Entity siswa → field = Nim, nama_siswa,alamat,dll
   Entity nasabah →field=Kd_nasabah,nama_nasabah,dll
d. Data value yaitu data aktual atau informasi yang disimpan pada tiap data elemen atau atribute.
   Contoh :
   Atribut nama_karyawan →sutrisno, budiman, dll
e. Record/tuple yaitu kumpulan elemen-elemen yang saling berkaitan menginformasikan tentang suatu entity secara lengkap.
f. File yaitu kumpulan record-record sejenis yang mempunyai panjang elemen sama, atribute yang sama namun berbeda-beda data valuenya.
g. Kunci elemen data yaitu tanda pengenal yang secara unik mengindentifikasikan entitas dari suatu kumpulan entitas



Gambar 1.1 Contoh Penggambaran Suatu Entity

Analisa Kasus

Kasus Pertama

- Perpustakaan Smart adalah perpustakaan umum yang anggotanya pelajar, mahasiswa dan masyarakat yang didirikan oleh Walikota Jakarta Barat. Keberadaan perpustakaan berlokasi di Walikota yang aplikasi pelayanan masih bersifat tradisional.
- Prosesnya :
  a. Setiap calon anggota yang akan menjadi anggota harus mengisi formulir dengan biaya administrasi Rp.10.000,-
  b. Anggota dapat meminjam buku maksimal 3 buku
  c. Untuk masa peminjaman selama 1 minggu (7 hari)
  d. Keterlambatan pengembalian dikenakan denda
  e. sesuai dengan kondisi denda,

Diantaranya :

1. Denda keterlambatan pengembalian dikenakan biaya administrasi Rp.500 perharinya (bukti surat denda terlampir)
2. Denda Buku perpustakaan rusak maka dikenakan biaya revisi buku perpustakaan(biaya ini dikenakan setelah buku diperbaiki).(bukti surat denda terlampir)
3. Denda Buku Hilang, maka dikenakan biaya penggantian seharga buku tersebut.(bukti surat denda terlampir)
4. Perpustakaan smart dapat menerima sumbangan dari donatur statusnya (anggota atau masyrakat luas)

Kasus Kedua

Sistem Informasi Inventaris

Suatu perusahaan software diminta membuatkan basis data yang akan menangani data-data inventaris pada sebuah toko. Karena tokonya kecil, maka ada beberapa gudang yang khusus untuk menyimpan stock produk. Data-data yang akan ditanganinya adalah: data produk yang ditawarkan toko, data pemasok produk, data transaksi pembelian produk dari pemasok (nota pembelian), dan data gudang tempat penyimpanan produk. Satu produk yang sama bisa disimpan di beberapa gudang yang berbeda, dan tentu saja tiap gudang menyimpan berbagai macam produk. Di database harus ada data mengenai sisa stock yang ada di masing-masing gudang untuk semua produk.

BAB II
Database Management System (DBMS) & Perancangan Basis Bata

2.1 Database Management System (DBMS)

DBMS adalah perangkat lunak yang memungkinkan pemakai untuk mendefinisikan, mengelola, dan mengontrol akses ke basis data. DBMS yang mengelola basis data relational disebut dengan Relational DBMS (**RDBMS**)
Contoh perangkat lunak yang termasuk DBMS: dBase, FoxBase, Rbase, Microsoft-Access, Borland Paradox / Borland Interbase, MS-SQL Server, Oracle, Informix, Sybase, MySQL, dll.

Bahasa dalam DBMS
Structure Query Language (SQL) adalah bahasa standar basis data yang digunakan aplikasi atau pemakai untuk berinteraksi dengan basis data melalui DBMS.
SQL dibagi menjadi dua, yaitu:

1. Data Definision Language (DDL)
   SQL yang digunakan untuk mendefinisikan basis data.
2. Data Manipulation Language (DML)
   SQL yang digunakan untuk mengkases dan mengelola data pada basis data.

Data Definition Language (DDL)
Dalam bahasa ini dapat membuat tabel baru, membuat indeks, menentukan struktur penyimpanan tabel, dan sebagainya. Hasil kompilasi perintah DDL disimpan dalam *file* khusus yang disebut **Kamus Data** (*Data Dictionary*).
Kamus Data merupakan suatu metadata (super-data) yaitu data yang mendeskripsikan data sesungguhnya.

Data Manipulation Language (DML)
Bahasa yang berguna untuk melakukan manipulasi data pada suatu basis data. Manipulasi dapat berupa: penambahan, penghapusan, pengubahan data pada suatu basis data.
Ada dua tipe DML, yaitu:

1. Prosedural, bahasa yang mensyaratkan pemakai untuk menentukan data apa yang diinginkan serta bagaimana cara untuk mendapatkannya.
2. Non Prosedural, bahasa yang membuat pemakai dapat menentukan data apa yang diinginkan tanpa menyebutkan bagaimana cara untuk mendapatkannya.

Komponen DBMS

1. **Query Prosesor**, komponen yang mengubah bentuk query kedalam instruksi kedalam database manager
2. **Database Manager**, menerima query & menguji eksternal & konceptual untuk menentukan apakah record – record tersebut dibutuhkan untuk memenuhi permintaan kemudian database manager memanggil file manager untuk menyelesaikan permintaan

3. **File Manager**, memanipulasi penyimpanan file dan mengatur alokasi ruang penyimpanan disk
4. **DML Precompiler**, modul yang mengubah perintah DML yang ditempelkan kedalam program aplikasi dalam bentuk fungsi-fungsi
5. **DDL Compiler**, merubah statement DDL menjadi kumpulan table atau file yang berisi data dictionary / meta data
6. **Dictionary Manajer**, mengatur akses dan memelihara data dictionary

Keuntungan DBMS:

- Mengurangi pengulangan data
- Mencapai independensi data
- Mengintegrasikan data beberapa file
- Mengambil data dan informasi dengan cepat
- Meningkatkan keamanan

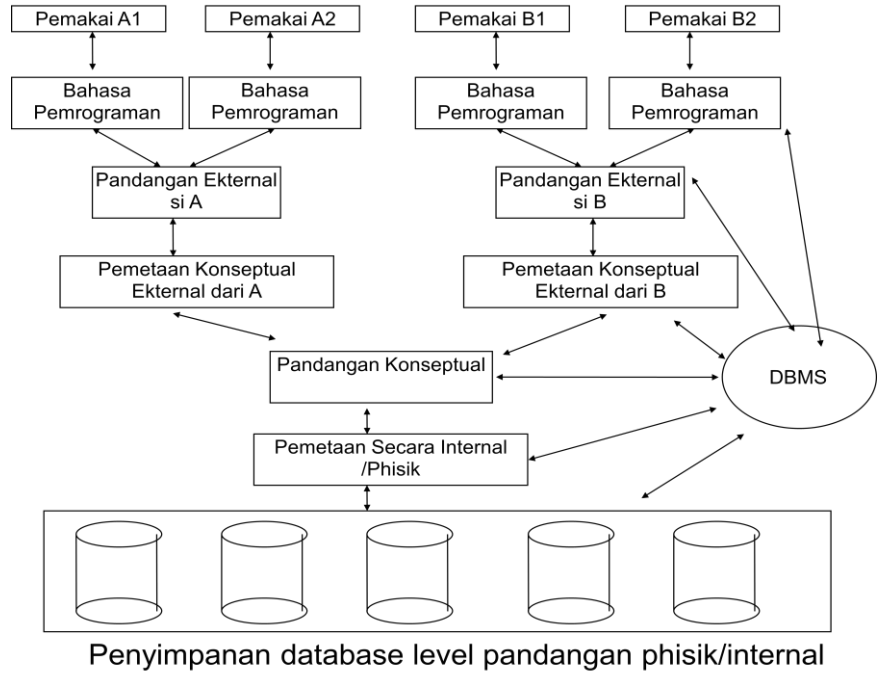
Kerugian DBMS:

- Perangkat lunak yang mahal
- Konfigurasi perangkat keras yang besar
- Mempekerjakan dan mempertahankan staf DBA

Abstraksi Data

Terbagi menjadi tiga tingkatan :

1. Internal level yaitu menerangkan struktur penyimpanan basisdata secara fisik dan organisasi file yang digunakan “
2. Konseptual level yang menerangkan secara menyeluruh dari basisdata dengan menyembunyikan penyimpanan data secara fisik “
3. Ekternal level yang menerangkan View basisdata dari sekelompok pemakai



Gambar 2.1 Arsitektur Sistem Database

2.2 Perancangan Basis Bata

Tujuan Perancangan Basis Data:

1. Untuk memenuhi informasi yang berisi kebutuhan–kebutuhan user secara khusus dan aplikasinya.
2. Memudahkan pengertian struktur informasi
3. Mendukung kebutuhan–kebutuhan pemrosesan dan beberapa objek penampilan (*respone time, processing time* dan *strorage space*)

Fase Perancangan Basis Data

Ada 6 fase proses perancangan database:

1. Pengumpulan dan analisa
   a. Menentukan kelompok pemakai dan bidang-bidang aplikasinya
   b. Peninjauan dokumentasi yang ada
   c. Analisa lingkungan operasi dan pemrosesan data
   d. Daftar pertanyaan dan wawancara
2. Perancangan database secara konseptual
   a. Perancangan skema konseptual
   b. Perancangan transaksi yang akan terjadi dalam database.
3. Pemilihan DBMS
   a. Faktor teknis
      Contoh faktor teknik :
      Tipe model data (hirarki, jaringan atau relasional), Struktur penyimpanan dan jalur pengaksesan yang didukung sistem manajemen database, Tipe interface dan programmer, Tipe bahasa query
   b. Faktor Ekonomi dan Politik organisasi
      Biaya penyediaan hardware dan software, Biaya konversi pembuatan database, Biaya personalia, dll
4. Perancangan database secara logik (data model mapping)
   a. Pemetaan (Transformasi data)
      Transformasi yang tidak tergantung pada sistem, pada tahap ini transformasi tidak mempertimbangkan karakteristik yang spesifik atau hal– hal khusus yang akan diaplikasikan pada sistem manajemen database
   b. Penyesuaian skema ke DBMS
      Penyesuaian skema yang dihasilkan dari tahap Pemetaan untuk dikonfirmasikan pada bentuk implementasi yang spesifik dari suatu model data seperti yang digunakan oleh sistem manajemen database yang terpilih
5. Perancangan database secara fisik
   a. Response Time
      Waktu transaksi database selama eksekusi untuk menerima respon
   b. Space Utility
      Jumlah ruang penyimpanan yang digunakan oleh database file dan struktur jalur pengaksesannya

c. Transaction Throughput

Merupakan nilai rata–rata transaksi yang dapat di proses permenit oleh sistem database dan merupakan parameter kritis dari sistem transaksi

6. Phase Implementasi Sistem Database

BAB III
Model Data

3.1 Model Data

Pengertian Model Data :

Sekumpulan konsep-konsep untuk menerangkan data, hubungan-hubungan antara data dan batasan-batasan data yang terintegrasi di dalam suatu organisasi.

Jenis-Jenis Model Data

A. Model Data Berdasarkan Object
B. Model Data Berdasarkan Record

A. Model Data Berbasis Objek

Model data berbasis objek menggunakan konsep entitas, atribut dan hubungan antar entitas. Terdiri dari:

1. Model Keterhubungan Entitas (*Entity-Relationship Model*)
2. Model Berorientasi Object (*Object-Oriented Model*)
3. Model Data Semantik (*Semantic Data Model*)
4. Model Data Fungsional (*Functional Data Model*)

Model Keterhubungan Entitas (*Entity-Relationship Model*) merupakan model yang paling populer digunakan dalam perancangan basis data.

Entity Relationship Model

Model untuk menjelaskan hubungan antar data dalam basis data berdasarkan suatu persepsi bahwa real word terdiri dari objek-object dasar yang mempunyai hubungan atau relasi antara objek-objek tersebut.

Komponen utama pembentuk Model Entity-Relationship, yaitu: **Entitas** (*Entity*), **Relasi** (Relation). Kedua komponen ini dideskripsikan lebih lanjut melalui sejumlah **Atribut/Properti**.

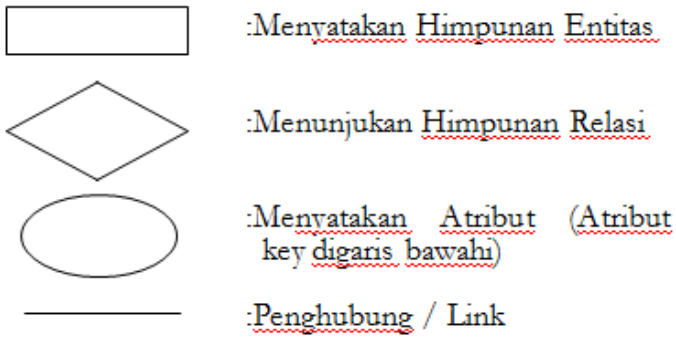
Diagram *Entity-Relationship* (Diagram E-R)

Model Entity Relationship yang berisi komponen himpunan entitas, relasi, yang dilengkapi atribut-atribut, dapat digambarkan menggunakan Diagram *Entity-Relationship* (Diagram E-R).

Menurut (Prasojo, 2014) Model ERD (*entity relationship diagram*) dibuat berdasarkan anggapan bahwa dunia nyata terdiri dari koleksi objek-objek dasar yang dinamakan entitas (*entity*) serta hubungan (*relationship*) antara entitas-entitas itu. Entitas adalah "sesuatu" atau "objek" pada dunia nyata yang dapat dibedakan antara satu dengan yang lainnya, yang bermanfaat bagi aplikasi yang akan kembangkan. Sebagai contoh, setiap **orang** adalah entitas dan **rekening bank** dapat dipertimbangkan sebagai sebuah entitas.

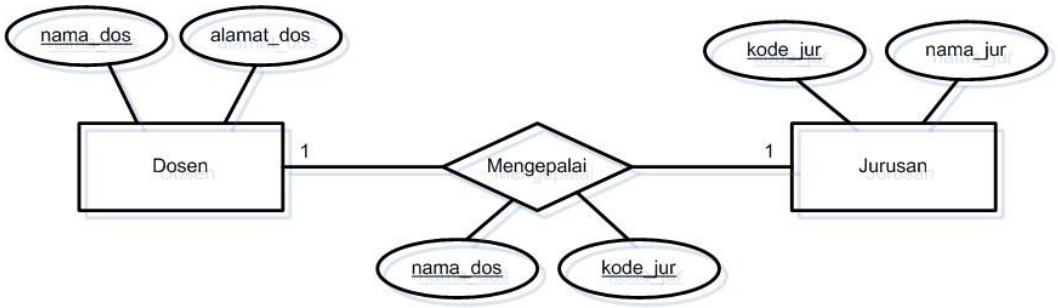
Entitas dalam *database* dideskripsikan berdasarkan atributnya. Sebagai contoh, **nomor rekening** membedakan suatu rekening adalah milik seseorang yang menyimpan uangnya dengan rekening milik orang lain di suatu bank tertentu dan nomor-nomor rekening tersebut merupakan atribut dari entitas rekening yang bersangkutan. Dalam hal ini, nomor rekening secara unik membedakan sebuah rekening dengan rekening yang lainnya.

Beberapa rekening mungkin memiliki saldo yang sama, tetapi mereka pasti memiliki nomor rekening yang berbeda. *Relationship* adalah hubungan antara beberapa entitas.
Simbol dasar yang digunakan :

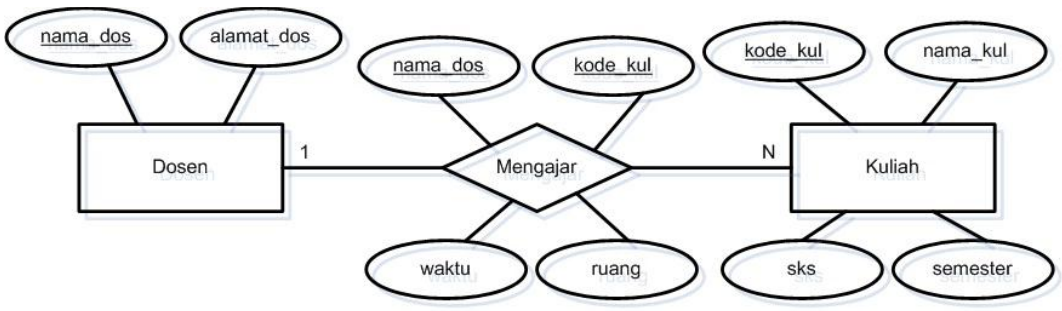


Dalam Diagram E-R aturan terpenting adalah Kardinalitas relasi/**Mapping Cardinalities** yang menentukan jumlah entity yang dapat dikaitkan dengan entity lainnya melalui relationship-set.
Jenis Mapping Cardinalities:

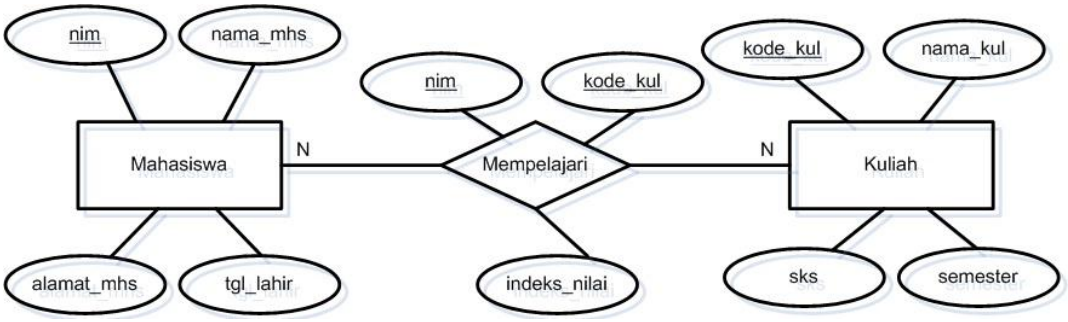
- Relasi satu ke satu (*one-to-one*)
- Relasi satu ke banyak (*one-to-Many*)
- Relasi banyak ke banyak (*many-to-many*)



Gambar 3.1 Contoh Relasi *one-to-one*



Gambar 3.2 Contoh Relasi *one-to-many*



Gambar 3.3 Contoh Relasi *many-to-many*

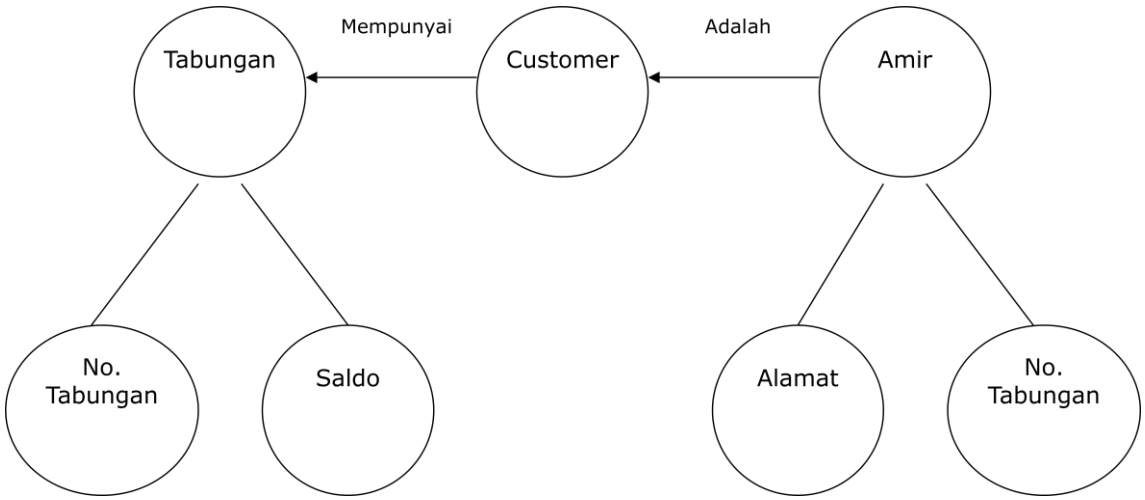
Semantic Model

Hampir sama dengan Entity Relationship model dimana relasi antara objek dasar tidak dinyatakan dengan simbol tetapi menggunakan kata-kata (Semantic). Sebagai contoh, dengan masih menggunakan relasi pada Bank X sebagaimana contoh sebelumnya, dalam semantic model adalah seperti terlihat pada gambar di atas.

Tanda-tanda yang menggunakan dalam semantic model adalah sebagai berikut :

: Menunjukkan adanya relasi

: menunjukkan atribut



Gambar 3.4 Contoh Kasus Semantic Model

B. Model Data Berbasis Record

Model ini berdasarkan pada record untuk menjelaskan kepada user tentang hubungan logic antar data dalam basis data

Perbedaan Dengan Model Data Berbasis Objek

Pada record based data model disamping digunakan untuk menguraikan struktur logika keseluruhan dari suatu database, juga digunakan untuk menguraikan implementasi dari sistem database (higher level description of implementation)

Model Relational

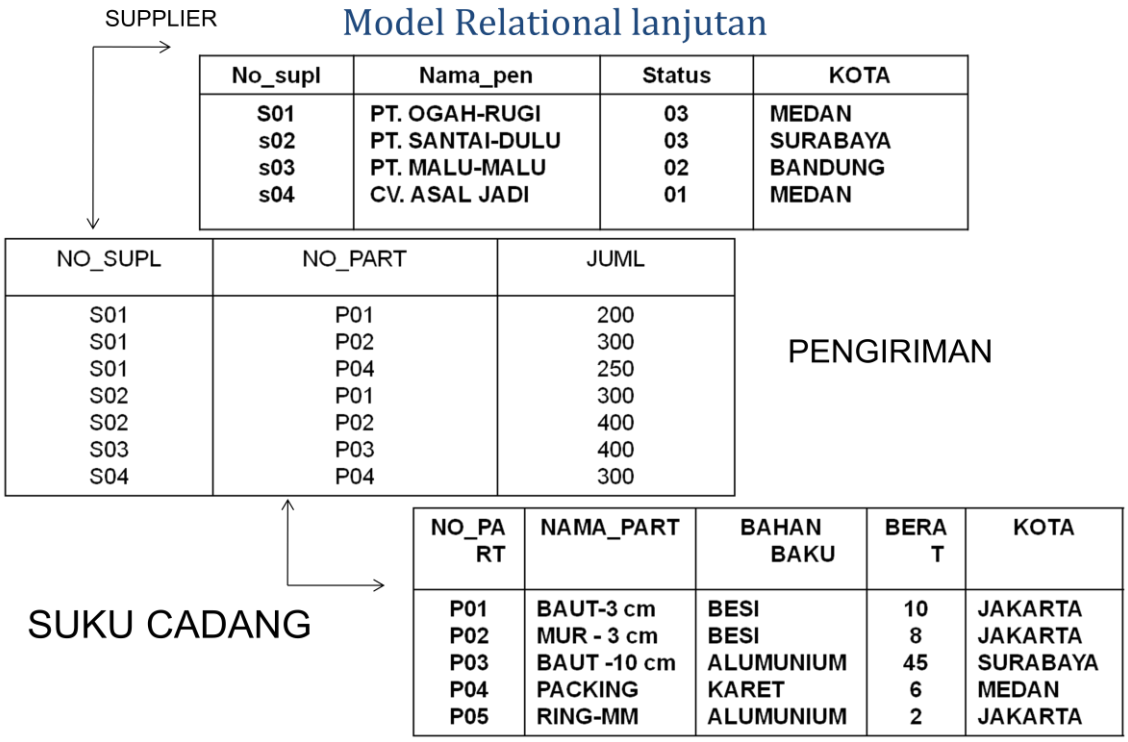
Terdapat 3 data model pada model data berbasis record:

1. Model Relational,

Dimana data serta hubungan antar data direpresentasikan oleh sejumlah tabel dan masing-masing tabel terdiri dari beberapa kolom yang namanya unique. Model ini berdasarkan notasi teori himpunan (set theory), yaitu relation.

Contoh : data base penjual barang terdiri dari 3 tabel:

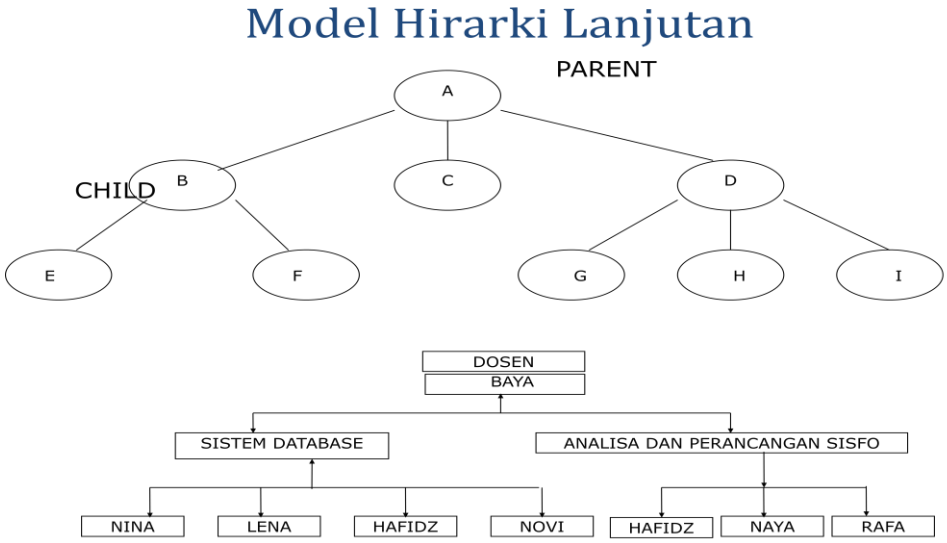
- Supllier
- Suku_cadang
- Pengiriman



Model Hirarki

2. Model Hirarki

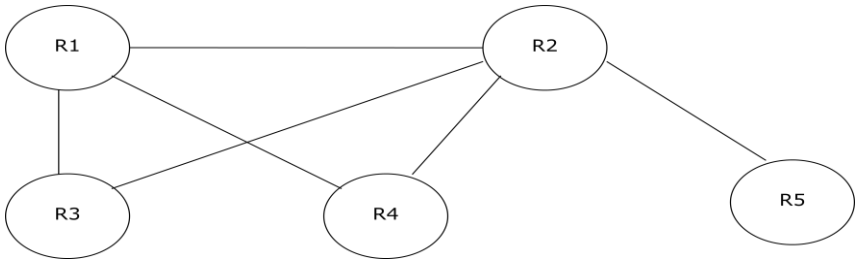
Dimana data serta hubungan antar data direpresentasikan dengan record dan link (pointer), dimana record-record tersebut disusun dalam bentuk tree (pohon), dan masing-masing node pada tree tersebut merupakan record/grup data elemen dan memiliki hubungan cardinalitas 1:1 dan 1:M .



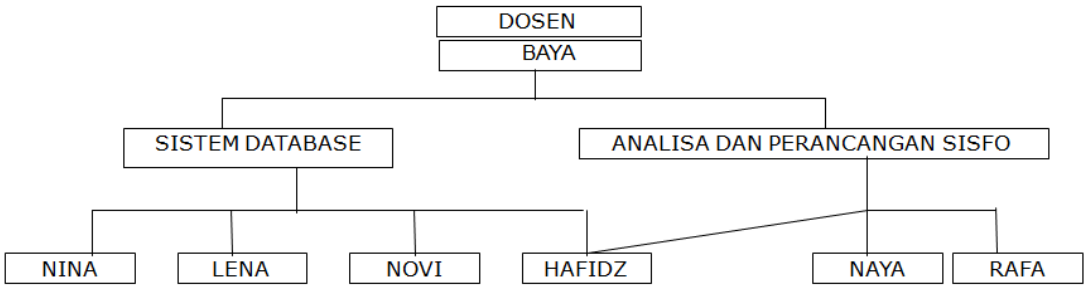
Model Jaringan

3. Model Jaringan

Distandarisasi tahun 1971 oleh Database Task Group (DBTG) atau disebut juga model CODASYL (Conference on Data System Language), mirip dengan hirarkical model dimana data dan hubungan antar data direpresentasikan dengan record dan links. Perbedaannya terletak pada susunan record dan linknya yaitu network model menyusun record-record dalam bentuk graph dan menyatakan hubungan cardinalitas 1:1, 1:M dan N:M.

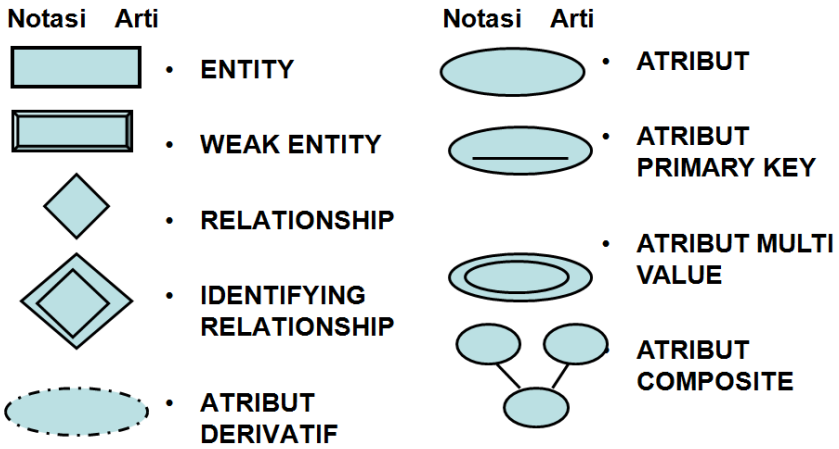


Model Jaringan lanjutan



BAB IV
Entity- Relationship diagram (ERD)

Simbol-simbol dalam E-R Diagram



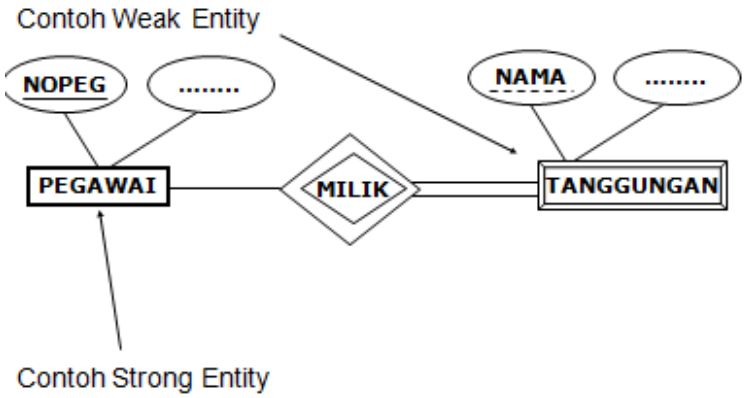
Komponen E-R Diagram

1. Entitas yaitu suatu kumpulan object atau sesuatu yang dapat dibedakan atau dapat diidentifikasikan secara unik. Dan kumpulan entitas yang sejenis disebut dengan entity set.
2. Relationship yaitu hubungan yang terjadi antara satu entitas atau lebih.
3. Atribut, kumpulan elemen data yang membentuk suatu entitas.
4. Indicator tipe terbagi 2 yaitu :
   a. Indicator tipe asosiatif object
   b. Indicator tipe super tipe

Entity Set

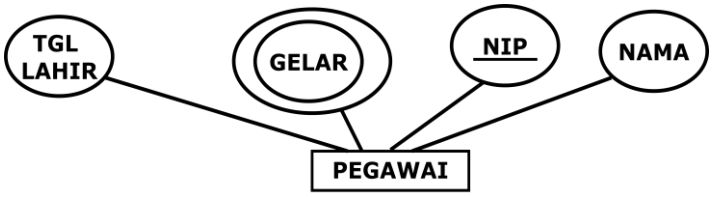
Entity Set Terbagi Atas :

1. Strong entity set yaitu entity set yang satu atau lebih atributnya digunakan oleh entity set lain sebagai key. Digambarkan dengan empat persegi panjang.
   Misal :
   E adalah sebuah entity set dengan atribute-atribute a1, a2,..,an, maka entity set tersebut direpresentasikan dalam bentuk tabel E yang terdiri dari n kolom, dimana setiap kolom berkaitan dengan atribute-atributenya.
2. Weak Entity set, Entity set yang bergantung terhadap strong entity set. Digambarkan dengan empat persegi panjang bertumpuk.
   Misal :
   A adalah weak entity set dari atribute-atribute a1, a2, .., ar dan B adalah strong entity set dengan atribute-atribute b1, b2,..,bs, dimana b1 adalah atribute primary key, maka weak entity set direpresentasikan berupa table A, dengan atribute-atribute {b1} u {a1,a2,.., ar}



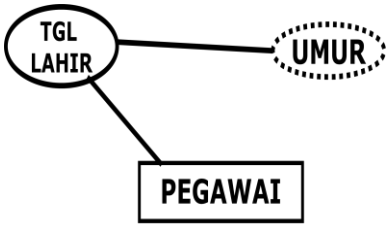
Jenis-Jenis Atribut

a. KEY → atribut yang digunakan untuk menentukan suatu entity secara unik
b. ATRIBUT SIMPLE →atribut yang bernilai tunggal
c. ATRIBUT MULTI VALUE →atribut yang memiliki sekelompok nilai untuk setiap instan entity
   Pada gambar dibawah ini, yang menjadi atribut key adalah NIP.
   Tgl Lahir dan Nama adalah atribut simple. Sedangkan Gelar merupakan contoh atribut multivalue.



d. ATRIBUT COMPOSIT →Suatu atribut yang terdiri dari beberapa atribut yang lebih kecil yang mempunyai arti tertentu contohnya adalah atribut nama pegawai yang terdiri dari nama depan, nama tengah dan nama belakang.

e. ATRIBUT DERIVATIF →Suatu atribut yg dihasilkan dari atribut yang lain. Sehingga umur yang merupakan hasil kalkulasi antara Tgl Lahir dan tanggal hari ini. Sehingga keberadaan atribut umur bergantung pada keberadaan atribut Tgl Lahir.



Mapping Cardinality

Banyaknya entity yang bersesuaian dengan entity yang lain melalui relationship

Jenis-Jenis Mapping :

1. One to one
2. Many to One atau One to many
3. Many to many

Representasi Dari Entity Set

Entity set direpresentasikan dalam bentuk tabel dan nama yang unique. Setiap tabel terdiri dari sejumlah kolom, dimana masing-masing kolom diberi nama yang unique pula

Participation Constraint

Menjelaskan apakah keberadaan suatu entity tergantung pada hubungannya dengan entity lain.

Terdapat dua macam participation constrain yaitu:

1. Total participation constrain yaitu:

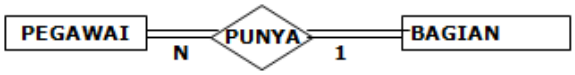
   Keberadaan suatu entity tergantung pada hubungannya dengan entity lain. Didalam diagram ER digambarkan dengan dua garis penghubung antar entity dan relationship.

2. Partial participation, yaitu

   Keberadaan suatu entity tidak tergantung pada hubungan dengan entity lain. Didalam diagram ER digambarkan dengan satu garis penghubung.

Contoh Participation Constraint

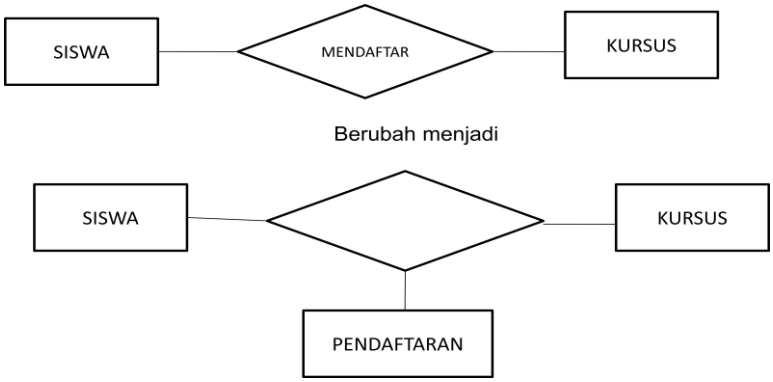
a. TOTAL PARTICIPATION



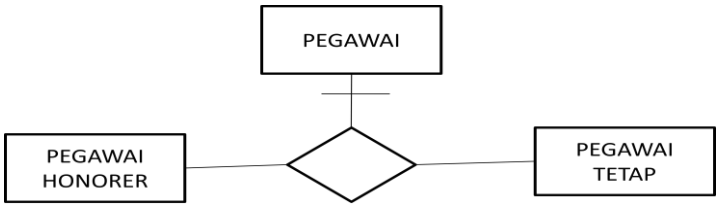
b. PARTIAL PARTICIPATION

Indicator Tipe

Indicator tipe asosiatif object berfungsi sebagai suatu objek dan suatu relationship.



Indicator tipe super tipe, terdiri dari suatu object dan satu subkategori atau lebih yang dihubungkan dengan satu relationship yang tidak bernama.



Logical Record Structured (LRS)

LRS → representasi dari struktur record-record pada tabel-tabel yang terbentuk dari hasil relasi antar himpunan entitas.

Menentukan Kardinalitas, Jumlah Tabel dan Foreign Key (FK)

One to One (1-1)

Gambar di atas menunujukan relasi dengan kardinalitas 1-1, karena:

1 supir hanya bisa mengemudikan 1taksi, dan

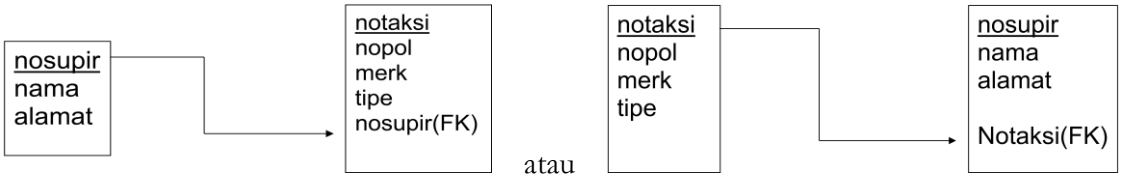
1 taksi hanya bisa dikemudikan oleh 1 supir.

Relasi 1-1 akan membentuk 2 tabel:

Tabel Supir (nosupir, nama, alamat)

Tabel Taksi (notaksi, nopol, merk, tipe)

LRS yang terbentuk sbb:



One to Many (1-M)

Gambar di atas menunujukan relasi dengan kardinalitas 1-M, karena:

1 Dosen bisa membimbing banyak Kelas, dan

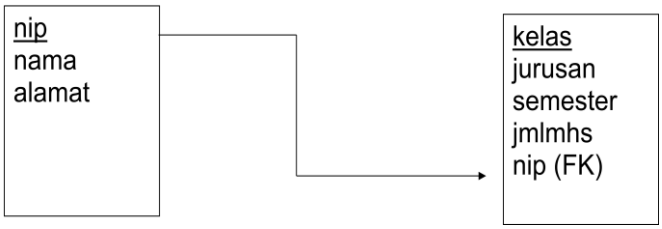
1 Kelas hanya dibimbing oleh 1 Dosen.

Relasi 1-M akan membentuk 2 tabel:

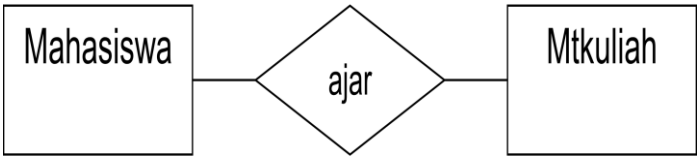
Tabel Dosen (nip, nama, alamat)

Tabel Kelas (kelas, jurusan, semester, jmlmhs)

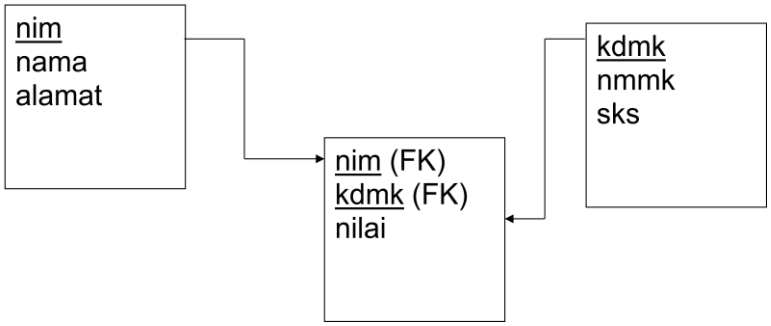
LRS yang terbentuk sbb:



Many to Many (M-M)



Gambar di atas menunujukan relasi dengan kardinalitas M-M, karena:
1 Mahasiswa bisa belajar banyak Mata Kuliah, dan
1 Mata Kuliah bisa dipelajari oleh banyak Mahasiswa.
Relasi M-M akan membentuk 3 tabel:
Tabel Mahasiswa (nim, nama, alamat)
Tabel Mtkuliah (kdmk, nmmk, sks)
Tabel Nilai (nim, kdmk, nilai) → menggunakan super key/composite key
LRS yang terbentuk sbb:



BAB V
Implementasi ERD dan LRS

Contoh Kasus

Sebuah perusahaan mempunyai beberapa bagian. Masing-masing bagian mempunyai pengawas dan setidaknya satu pegawai. Pegawai harus ditugaskan pada paling tidak satu bagian, tetapi dapat pula beberapa bagian. Paling tidak satu pegawai mendapat tugas sebuah proyek. Namun, seorang pegawai dapat libur dan tidak mendapat tugas proyek.

Penyelesaian

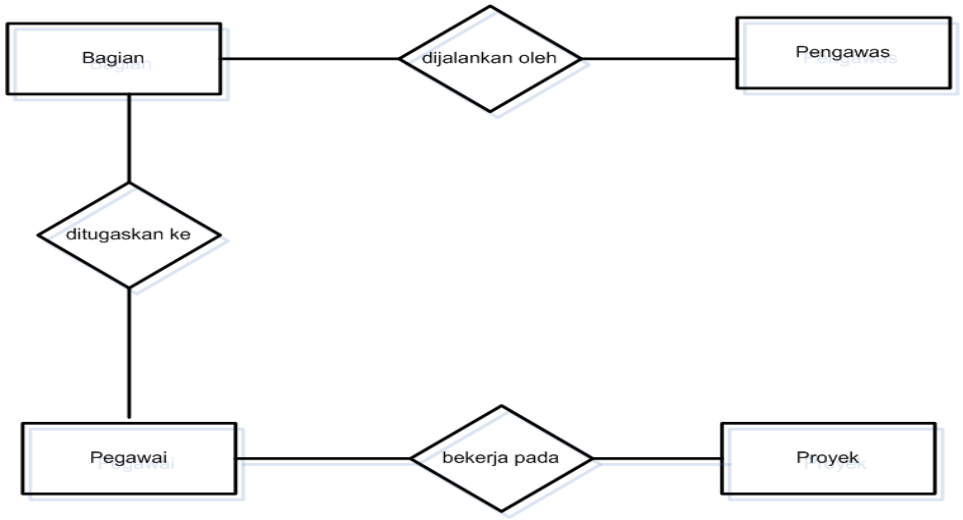
Langkah 1 : Menentukan Entitas

Entitas yang dibutuhkan adalah : Bagian, Pegawai, Pengawas, dan Proyek

Langkah 2 :Menentukan Relasi dengan matriks relasi

	Bagian	Pegawai	Pengawas	Proyek
Bagian		ditugaskan ke	dijalankan oleh	
Pegawai	milik			bekerja pada
Pengawas	menjalankan			
Proyek		menggunakan		

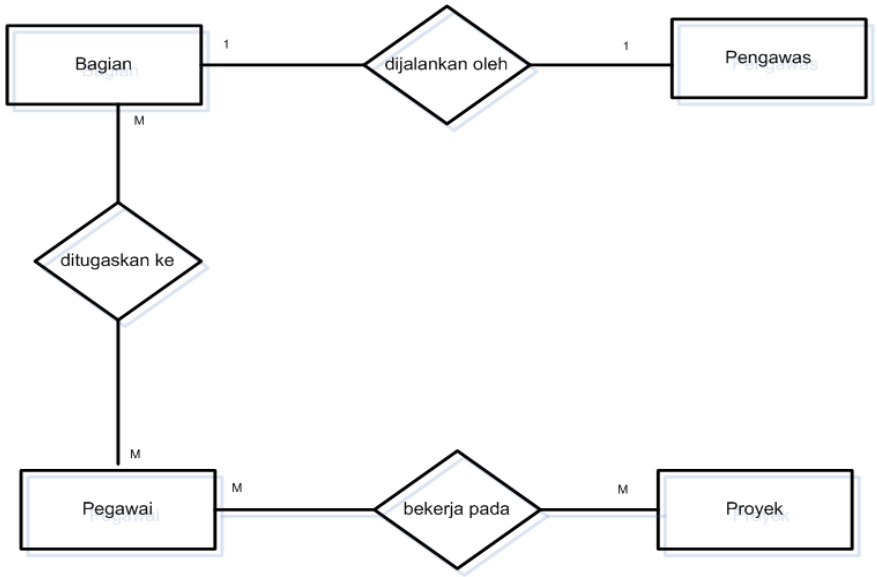
Langkah 3 : Menggambar ERD Sementara



Deskripsi Permasalahan :

- Masing-masing bagian hanya mempunyai satu pengawas
- Seorang pengawas hanya bertugas pada satu bagian
- Masing-masing bagian memiliki paling tidak satu pegawai
- Masing-masing pegawai bekerja paling tidak pada satu bagian
- Masing-masing proyek dikerjakan oleh paling tidak satu pegawai
- Seorang Pengawas bisa mendapat tugas 0 atau beberapa proyek

Langkah 4 : Mengisi Kardinalitas



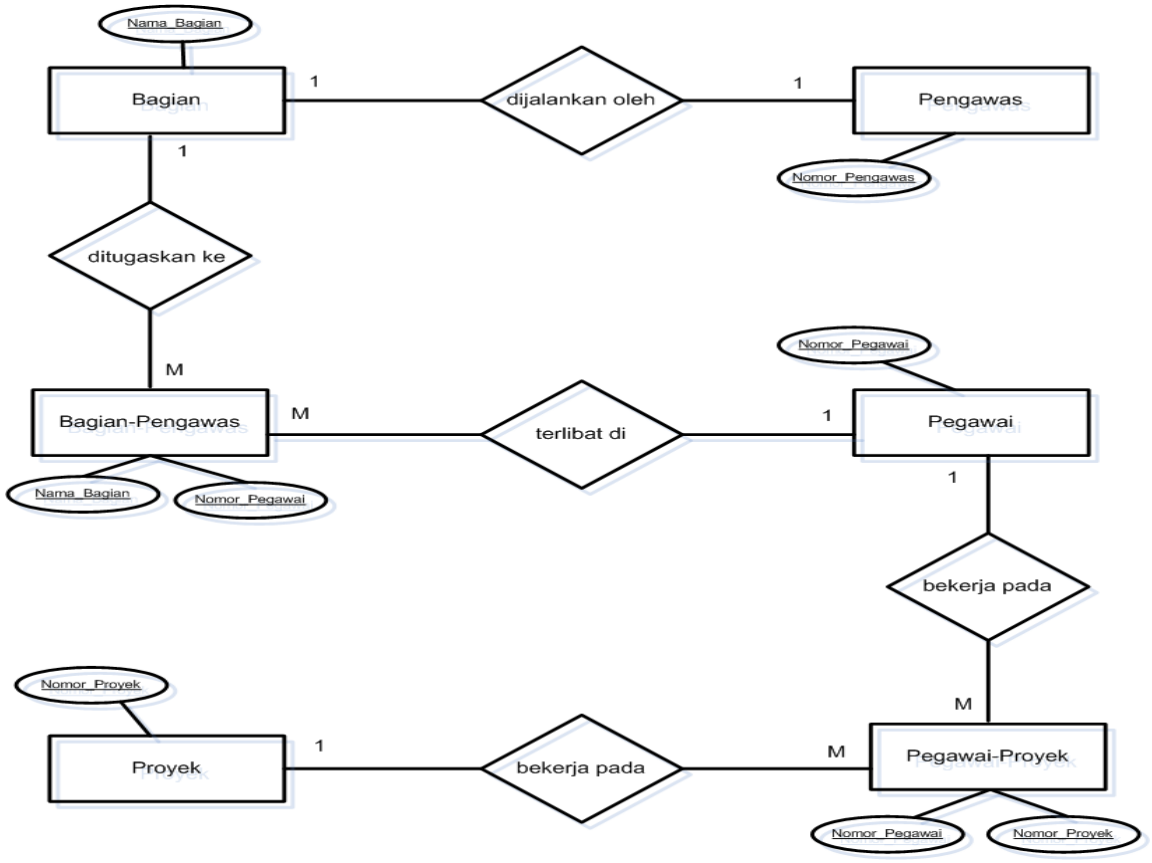
Langkah 5: Menentukan Kunci Utama

Kunci Utama : **Nama Bagian, Nomor Pengawas, Nomor Pegawai, Nomor Proyek.**

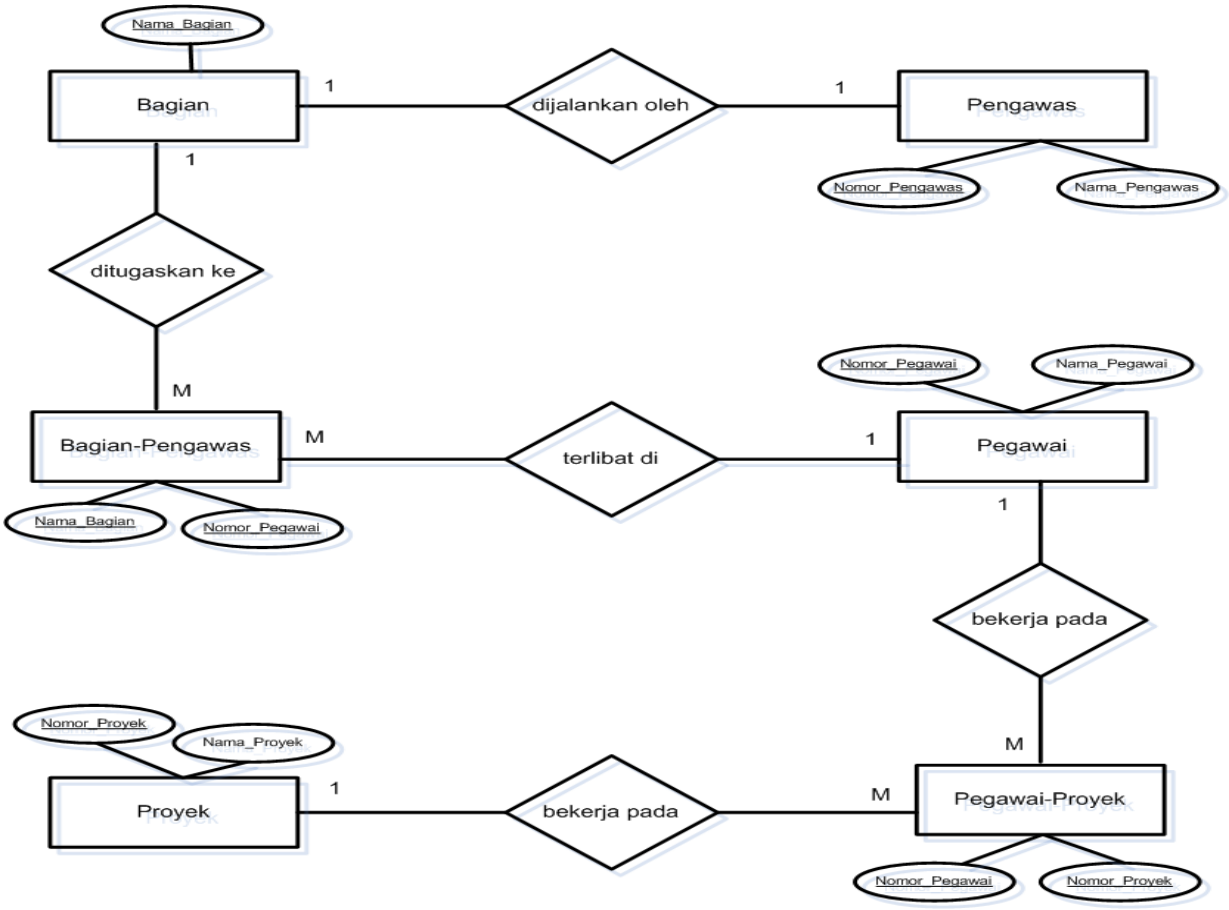
Langkah 6: Menggambarkan ERD berdasarkan kunci

Karena ada dua relasi many-to-many pada ERD sementara, yaitu antara Bagian dan Pegawai, serta Pegawai dan Proyek. Oleh karena itu dibuatkan entitas baru yaitu Bagian-Pegawai dan Pegawai-Proyek. Kunci utama Bagian-Pegawai adalah gabungan Nama Bagian dan Nomor Pegawai. Kunci utama Pegawai-Proyek adalah gabungan Nomor Pegawai dan Nomor Proyek

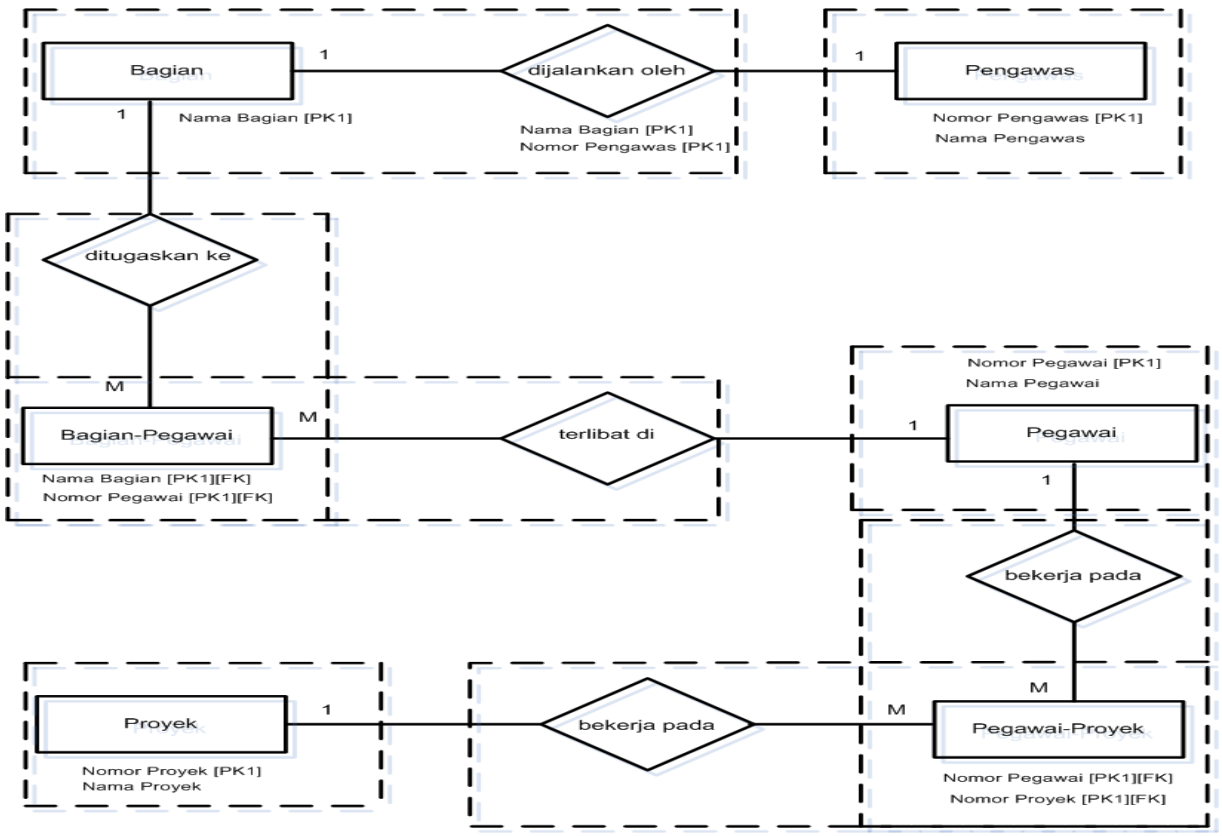
Penggambaran ERD Berdasarkan Kunci



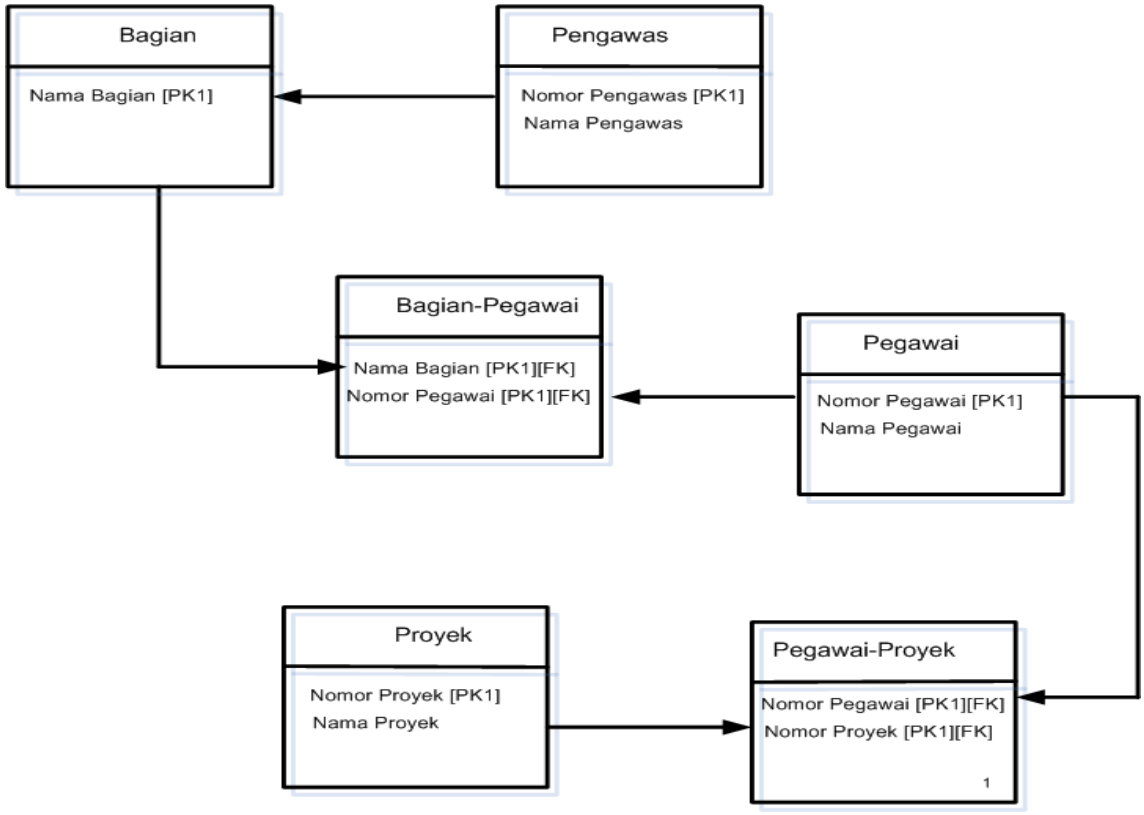
Langkah 7: Menentukan Atribut yang diperlukan



Transformasi ERD dan LRS



LRS yang terbentuk



Analisa Kasus ERD Perpustakaan Smart (Pembahasan di Kelas)

1. Pembuatan gambar ERD dari Perpustakaan Smart

Langkah –langkah pembuatan ERD dan LRS

Tentukan entity – entity yang diperlukan
Tentukan relationship antar entity – entity
Menggambar ERD Sementara
Mengisi kardinalitas
Menentukan kunci utama
Menggambar ERD Berdasarkan Kunci
Tentukan attribute – attribute
Transformasi ERD ke LRS
Menggambar LRS

BAB VI
Teknik Normalisasi

Beberapa Pengertian Normalisasi :

Normalisasi adalah suatu proses memperbaiki atau membangun dengan model data relasional, dan secara umum lebih tepat dikoneksikan dengan model data logika (Mulyanto, 2009).

Normalisasi adalah proses pengelompokan atribute-atribute dari suatu relasi sehingga membentuk WELL STRUCTURE RELATION.

Well Structure Relation

Adalah sebuah relasi yang jumlah kerangkapan datanya sedikit (*minimum Amount Of Redundancy*), serta memberikan kemungkinan bagi user untuk melakukan INSERT, DELETE, dan MODIFY terhadap baris-baris data pada relation tersebut, yang tidak berakibat terjadinya ERROR atau INKONSESTENSI DATA, yang disebabkan oleh operasi-operasi tersebut

Keuntungan Normalisasi

Keuntungan dari normalisasi, yaitu :

1. Meminimalkan ukuran penyimpanan yang diperlukan untuk menyimpan data.
2. Meminimalkan resiko inkonsistensi data pada basis data
3. Meminimalkan kemungkinan anomali pembaruan
4. Memaksimalkan stabilitas struktur data

ANOMALY

ANOMALY merupakan penyimpangan-penyimpangan atau Error atau inkonsistensi data yang terjadi pada saat dilakukan proses insert, delete maupun update.

Terdapat 3 jenis Anomaly :

1. Insertion Anomaly
   Error yang terjadi sebagai akibat operasi insert record/tuple pada sebuah relation
2. Deletion Anomaly
   Error yang terjadi sebagai akibat operasi delete record/tuple pada sebuah relation
3. Update Anomaly
   Error yang terjadi sebagai akibat inkonsistensi data yang terjadi sebagai akibat dari operasi update record/tuple dari sebuah relation

Problem-Problem Pada Relation yang Sudah Dinormalisasi

- Performance problem
  Masalah terhadap performa database
- Referential Integrity Problem
  Masalah yang timbul terhadap referensi antar data-data diantara dua tabel atau lebih

Beberapa Konsep Yang Harus Diketahui:

a. Field/ Atribut Kunci
b. Kebergantungan Fungsi

Atribut Kunci (Field)

Menurut (Hartono, 2005) menjelaskan bahwa terdapat bermacam-macam jenis *Key*, antara lain:

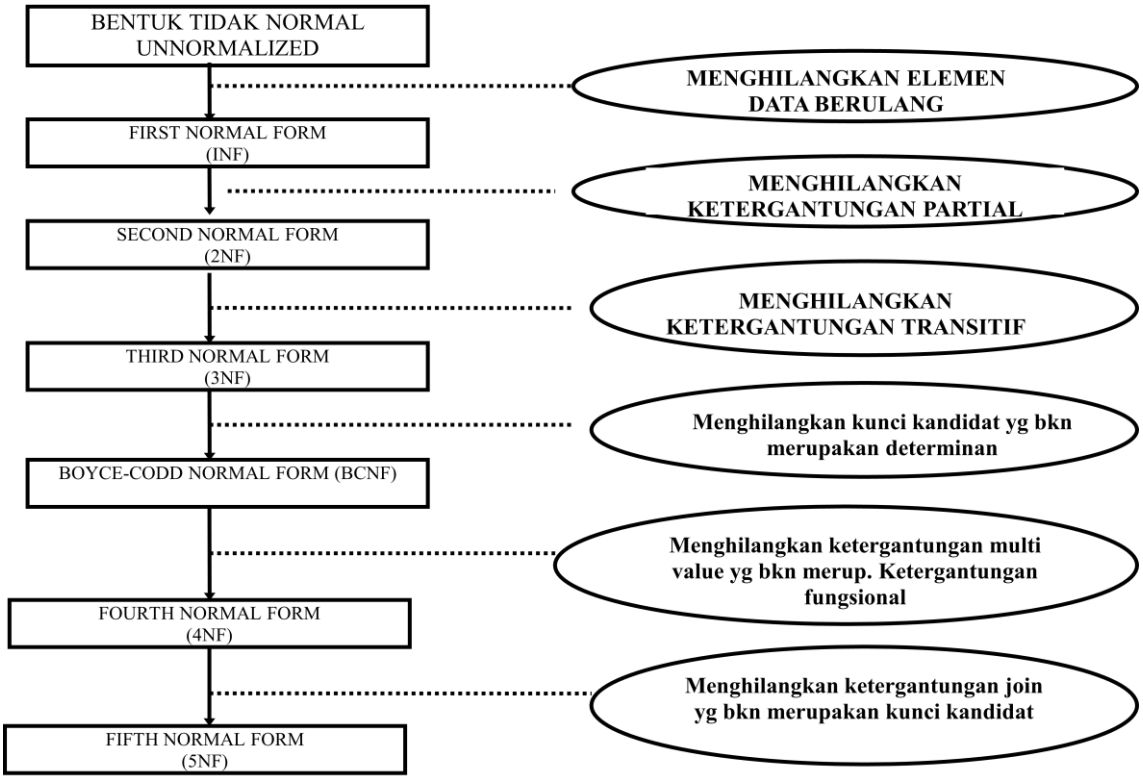
a. *Super Key*
   Salah satu atau lebih atribut (kumpulan atribut) dari suatu tabel yang dapat digunakan untuk mengidentifikasi *entity/record* dari tabel tersebut secara unik (tidak semua atribut dapat menjadi *super key*).
b. *Candidate Key*
   Tidak boleh berisi atribut dari tabel yang lain sehingga *candidate key* sudah pasti *super key* namun belum tentu sebaliknya.
c. *Primary Key*
   Salah satu atribut *candidate key* dapat dipilih menjadi *primary key* dengan 3 (tiga) kriteria yaitu *key* tersebut lebih natural untuk digunakan sebagai acuan, lebih sederhana, terjamin keunikannya.
d. *Alternate key*
   Setiap atribut *candidate key* yang tidak terpilih menjadi *primary key* maka atribut-atribut tersebut dinamakan *alternate key*.

Kebergantungan Kunci

1. Ketergantungan Fungsional (Fungsional Dependent)
   Keterkaitan antar hubungan antara 2 atribute pada sebuah relasi. Dituliskan dengan cara : A -> B, yang berarti :
   Atribute B fungsionality Dependent terhadap atribute A atau
   Isi (*value*) atribute A menentukan isi atribute B
   Definisi dari functional dependent :
   Diketahui sebuah relasi R, atribute Y dari R adalah FD pada atribute X dari R ditulis R.X -> R.Y jika dan hanya jika tiap harga X dalam R bersesuaian dengan tepat satu harga Y dalam R
2. Fully Functionaly Dependent (FFD)
   Suatu rinci data dikatakan fully functional dependent pada suatu kombinasi rinci data jika functional dependent pada kombinasi rinci data dan tidak functional dependent pada bagian lain dari kombinasi rinci data.
   Definisi dari FDD:
   Atribute Y pada relasi R adalah FFD pada atribute X pada relasi R jika Y FD pada X tida FD pada himpunan bagian dari X
   Contoh:
   PersonID,Project,Project_budget→time_spent_byperson_ onProject (bukan FFD)
   PersonID, Project →time_spent_byperson_onProject (FDD)
3. Ketergantungan Partial
   Sebagian dari kunci dapat digunakan sebagai kunci utama

4. Ketergantungan Transitif
   Menjadi atribute biasa pada suatu relasi tetapi menjadi kunci pada relasi lain
5. Determinan
   Suatu atribute (field) atau gabungan atribute dimana beberapa atribute lain bergantung sepenuhnya pada atribute tersebut

Langkah-Langkah Pembuatan Normalisasi



Studi Kasus Normalisasi

PT. SANTA PURI FAKTUR PEMBELIAN BARANG
Jalan senopati 11
yogyakarta

Kode Suplier : G01 Tanggal : 05/09/2000
Nama Suplier : Gobel Nustra Nomor: 998

Kode	Nama Barang	Qty	Harga	Jumlah
A01	AC SPLIT ½ PK	10.0	135,000	1,350,000
A02	AC SPLIT 1 PK	10.0	200,000	2,000,000
			Total Faktur	3,350,000

Jatuh tempo faktur : 09/09/2000

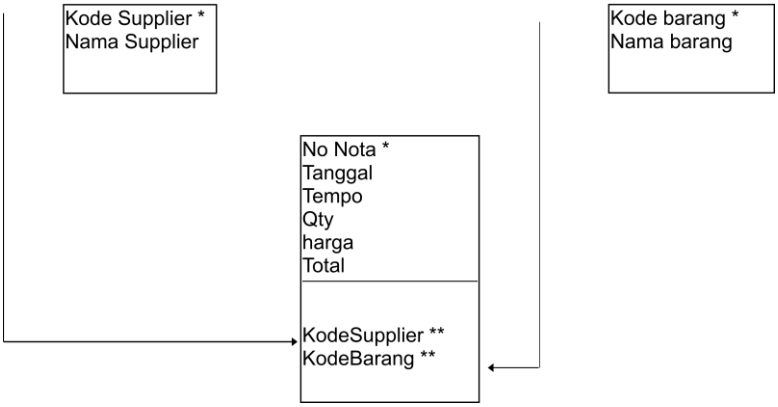
1. Step 1 bentuk unnormalized

no fac	kode supp	nama supp	kode brg	nama barang	tanggal	jatuh tempo	qty	harga	jumlah	Total
779 998	S02 G01	Hitachi Gobel N	R02 A01 A02	RICE COOKER AC SPLIT ½ PK AC SPLIT 1 PK	02/09/00 05/09/00	08/09/00 09/09/00	10 10 10	15000 135000 200000	150000 1350000 2000000	150000 3350000

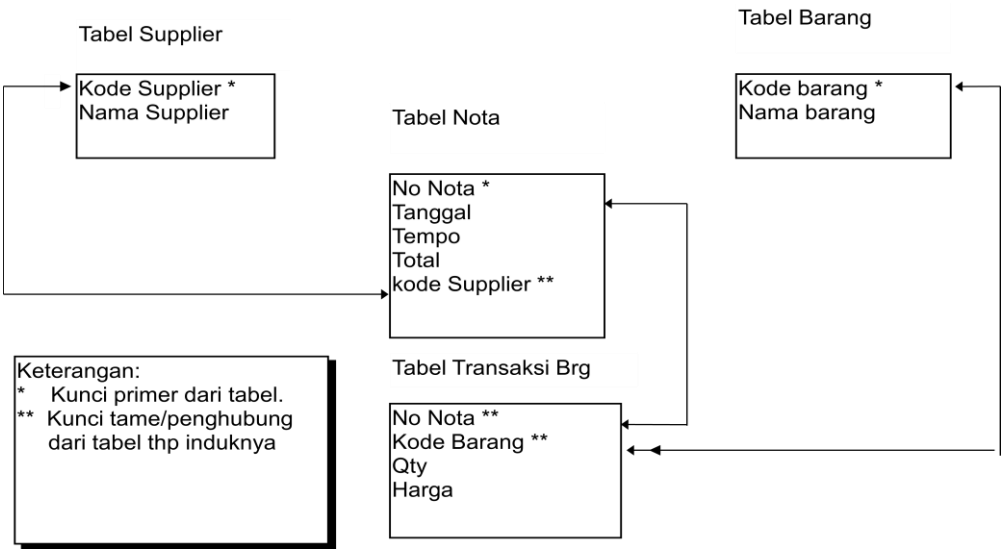
2. Step 2 bentuk 1 NF

nofac	kode supp	nama supp	Kode brg	nama barang	tanggal	jatuh tempo	qty	harga	jumlah	Total
779	S02	Hitachi	R02	RICE COOKER	02/09/00	08/09/00	10	15000	150000	150000
998	G01	Gobel N	A01	AC SPLIT ½ PK	05/09/00	09/09/00	10	135000	1350000	3350000
998	G01	Gobel N	A02	AC SPLIT 1 PK	05/09/00	09/09/00	10	200000	2000000	3350000

3. Step 3 bentuk 2 NF



4. Step IV Bentuk 3 NF



Latihan Soal

Buatlah bentuk Normalisasi dari dokumen berikut ini :

Kartu pengobatan masyarakat

No Pasien : 1234/PO/IV/99 **Tanggal Pendaftaran** : 1 Mei 1999

Data Pasien dari, **NOPEN** : 1000019999 **Nama Pasien** : Bachtiar Jose

Alamat Pasien, **Jalan** : Kebon Jeruk No. 27 **Kelurahan** : Palmerah

Kecamatan : Kemanggisan **Wilayah** : Jakarta Barat

Kode Pos :11530 **Telepon** : 5350999

NoRM	Tgl periksa	Kode Dokter	Nama Dokter	KodeSakit	Diskripsi sakit	Kode obat	Nama obat	Dosis
RM001	1/5/99	D01	Dr Zurmaini	S11	Tropicana	B01 B02	Sulfa Anymiem	3dd1 4dd1
RM002	4/7/99	D01	Dr Zurmaini	S12	Ulcer Triombis	B01 B03	Sulfa Supralin	3dd2 3dd1
RM003	4/4/99	D02	Dr Harjono	S12	Ulcer Triombis	B04	Adrenalin	4dd2
RM004	7/8/99	D04	Dr Mahendra	S12	Ulcer Triombis	B01 B02 B03	Sulfa Anymiem Supralin	3dd2 4dd2 3dd1

Penerima Pasien

Ttd

(Udin)

Buatlah bentuk un normal form, 1NF dan 2NF

BAB VII
Bahasa Query Formal

Dalam bahasa Query Formal, ada dua dasar pembentukan bahasa Query, yaitu:

1. Aljabar Relasional
2. Kalkulus Relasional

Dalam pembahasan ini hanya akan membahas tentang Aljabar Relasional karna lebih banyak dijadikan dasar Bahasa Query yang umum digunakan.

Aljabar Relasional

Adalah kumpulan operasi terhadap relasi, dimana setiap operasi menggunakan satu atau lebih relasi untuk menghasilkan satu relasi yang baru.

Bahasa *Query* yang didasarkan pada operasi-operasi dalam Aljabar Relasional merupakan bahasa *query* yang **Prosedural**.

B. OPERATOR RELATIONAL

1. Restrict (σ) adalah Pemilihan tupel atau record
2. Project (π) adalah pemilihan attribute atau field
3. Divide ($\div$) adalah membagi
4. Join (θ) adalah menggabungkan

ALJABAR RELASIONAL

Operator pada aljabar relational dibagi menjadi 2 kelompok:

1. Operator dasar untuk fundamental operational
2. Operator tambahan untuk additional operasional

Tabel dibawah ini adalah contoh untuk mengerjakan perintah – perintah Relation Algebra:

RELASI : MATA KULIAH

KD_MK	NAMA_MK	SKS	NIP
207	LOGIKA & ALGO	4	199910486
310	STRUKTUR DATA	3	200109655
360	SISTEM BASIS DATA	3	200209817
545	IMK	2	200209818
547	APSI	4	200109601
305	PEMR. PASCAL	4	200703073
544	DISAIN GRAFIS	2	200010490

RELASI : MAHASISWA

NIM	NAMA_MHS	ALAMAT	J_KEL
1105090222	HAFIDZ	DEPOK	LAKI-LAKI
1105091002	RAFFA	DEPOK	LAKI-LAKI
1105095000	NAIA	DEPOK	PEREMPUAN
1104030885	ARIF	P.LABU	LAKI-LAKI
1206090501	LENI	KMP. MELAYU	PEREMPUAN
1206090582	WAHYUNI	TANGERANG	PEREMPUAN
1205097589	ARIS	DEPOK	LAKI-LAKI
1106094586	YANI	CILEDUG	PEREMPUAN
110709	BAMBANG	SALEMBA	LAKI-LAKI

RELASI : REGISTRASI

KD_MK	NIM
360	1105090222
545	1206090501
547	1105095000

RELASI : DOSEN

NIP	NAMA_DOS	GAJI
199910486	BILLY	3500000
200109655	MARDIANA	4000000
200209817	INDRIYANI	4500000
200209818	SURYANI	4250000
200109601	DWINITA	3500000
200703073	MALAU	2750000
200010490	IRFIANI	3500000

Operator Dasar

a. Selection (σ) Lower Case Omega

Operasi selection menyeleksi tupel-tupel pada sebuah relation yang memenuhi predicate/syarat yang sudah ditentukan

Contoh :

1. Mencari tuple-tuple dari MAHASISWA yang memiliki jenis kelamin laki-laki, Ekspresi aljabar relational :

   σ J_KEL="LAKI-LAKI" (MAHASISWA)

2. Tampilkan data mata kuliah yang memiliki kode 360 atau yang memilki sks 4

   σ KD_MK="306" V SKS=4 (MATAKULIAH)

b. Projection (π)

Operator projection beroperasi pada sebuah relation, yaitu membentuk relation baru dengan mengcopy atribute-atribute dan domain-domain dari relation tersebut berdasarkan argumen-argumen pada operator tersebut.

Contoh :

Tampilkan nama beserta gaji dari dosen

$^{\pi}$nama_dos,gaji (DOSEN)

c. Cartesian product (X)

Operator dengan dua relasi untuk menghasilkan tabel hasil perkalian kartesian.

Contoh :

Tampilkan nid,nama_d (dari relasi Dosen), nama_mk (dari relasi Matakuliah), thn_akademik,smt,hari,jam_ke,waktu,kelas (dari relasi Mengajar) dimana semester mengajar adalah pada semester '1'.

π nid, nama_d, nama_mk, thn_akademik,smt, hari,jam_ke, waktu, kelas (σ smt=1 ∧Dosen.nid = Mengajar.nid ∧mengajar.kdmk = Matakuliah.kdmk (DosenxMatakuliahxMengajar))

d. Union (∪)

Operasi untuk menghasilkan gabungan tabel dengan syarat kedua tabel memiliki atribut yang sama yaitu domain atribut ke-i masing-masing tabel harus sama

RUS={ X I X E R atau X E S}

Contoh :

Penggabungan berdasarkan kolom kota dari tabel mahasiswa dengan tabel dosen

π kota (mahasiswa) ∪ πkota (Dosen)

e. Set diference (-)

Operasi untuk mendapatkan tabel dis uatu relasi tapi tidak ada di relasi lainnya.

R – S = { X I X E R dan X E S }

Contoh : Tampilkan nama dari mahasiswa yang tinggal di depok tetapi bukan berjenis kelamin perempuan

Query I : tampilkan nama yang tinggal di depok

πnama_mhs(σalamat="DEPOK" (MAHASISWA))

Query II : tampilkan nama yang berjenis kelamin perempuan

πnama_mhs(σj_kel ="PEREMPUAN" (MAHASISWA))

Tampilkan query I minus query II :

πnama_mhs(σalamat="DEPOK"(MAHASISWA))-πnama_mhs(σj_kel="PEREMPUAN" (MAHASISWA))

Operator Tambahan

1. **SET INTERSECTION (∩)**

   Operasi untuk menghasilkan irisan dua tabel dengan syarat kedua tabel memiliki atribut yang sama, domain atribut ke-i kedua tabel tersebut sama.

2. **THETA JOIN**

   Operasi yang menggabungkan operasi cartesian product dengan operasi selection dengan suatu kriteria.

3. **NATURAL JOIN**

   Operasi menggabungkan operasi selection dan cartesian product dengan suatu kriteria pada kolom yang sama

4. **DIVISION**

   Merupakan operasi pembagian atas tuple-tuple dari 2 relation

   Contoh:

Sno	Pno
S1	P1
S1	P2
S1	P3
S1	P4
S2	P1
S2	P2

A

B

Pno
P2

A/B

Sno
S1
S2

BAB VIII
Bahasa Query Terapan

***Structured Query Language* (SQL)**

SQL merupakan bahasa query terapan yang banyak digunakan oleh berbagai DBMS, diterapkan dalam berbagai *development tools* dan program aplikasi untuk berinteraksi dengan basis data.

Subdivisi SQL:

1. *Data Definition Language (DDL)*
   Query-query ini digunakan untuk mendefinisikan struktur atau skema basis data.
2. *Data Manipulation Language (DML)*
   Query-query ini digunakan untuk manajemen data dalam basis data.

PENGELOMPOKAN STATEMEN SQL

1. *Data Definition Language* (DDL)

CREATE DATABASE		DROP DATABASE
CREATE TABEL		DROP TABEL
CREATE	INDEX	DROP INDEX
CREATE	VIEW	DROP VIEW
ALTER TABLE		

2. *Data Manipulation Language* (DML)
   INSERT, SELECT, UPDATE, DELETE
3. Data Access
   GRANT , REVOKE
4. Data Integrity
   RECOVER TABLE
5. Auxiliary
   SELECT INTO OUTFILE, LOAD, RENAME TABLE

Data Definition Languange (DDL)

A. CREATE

1. **Pembuatan Database**

   Nama Database adalah yang dapat mewakili suatu kejadian dapat berupa nama organisasi atau perusahaan.

   Sintaks : CREATE DATABASE nama_database

   Contoh : Buat database dengan nama KAMPUS

   CREATE DATABASE KAMPUS

2. **Pembuatan Tabel**

   Sintaks : CREATE TABLE nama_table
   (nama_kolom1 tipe_data_kolom1,
   nama_kolom2,tipe_data_kolom2,….)

   Contoh :

   Buat struktur tabel dengan nama tabel Mahasiswa dengan data NIM char(8), NAMA char(25), ALAMAT char(30)

3. Pembuatan Index

Sintaks : CREATE [UNIQUE] INDEX nama_index
ON nama_table (nama_kolom) ;

Contoh :

Buat index data Mahasiswa berdasarkan NIM dengan nama MHSIDX Dimana NIM tidak boleh sama

CREATE UNIQUE INDEX MHSIDX ON Mahasiswa(NIM)

4. Pembuatan View

Sintaks :

CREATE VIEW nama_view [(nama_kolom1,….)]
AS SELECT statement
[WITH CHECK OPTION] ;

Contoh :

Buat view dengan nama MHSVIEW yang berisi semua data mahasiswa

CREATE VIEW MHSVIEW
AS SELECT * FROM Mahasiswa

CREATE TABLE Mahasiswa (NIM char(8) not null,
NAMA char(25) notnull, ALAMAT char(30) notnull)

B. DROP (MENGHAPUS)

1. Menghapus Database
   Sintaks : DROP DATABASE nama_db ;
2. Menghapus Tabel
   Sintaks : DROP TABLE nama_table ;
3. Menghapus Index
   Sintaks : DROP INDEX nama_index ;
4. Menhapus View
   Sintaks : DROP VIEW nama_view ;

Contoh :

DROP DATABASE **KAMPUS**;
DROP TABLE MHS;
DROP INDEX MHSIDX;
DROP VIEW MHSVIEW;

C. ALTER TABLE (MERUBAH STRUKTUR TABEL)

Sintaks: ALTER TABLE nama_tabel
ADD nama_kolom jenis_kolom
[FIRST | AFTER nama_kolom]
CHANGE [COLUMN] oldnama newnama
MODIFY nama_kolom jenis kolom, …
DROP nama_kolom
RENAME newnama_tabel

Contoh :

1. Tambahkan kolom JKEL dengan panjang 1 char pada tabel Mahasiswa
   ALTER TABLE Mahasiswa ADD JKEL char(1);
2. Ubah panjang kolom JKEL menjadi 15 char
   ALTER TABLE Mahasiswa MODIFY COLUMN JKEL char(15);
3. Hapus kolom JKEL dari data table MHS
   ALTER TABLE Mahasiswa DROP JKEL;

Data Manipulation Language (DML)

A. **INSERT**

Sintaks SQL yang digunakan untuk penambahan record baru kedalam sebuah tabel.
Sintaks: INSERT INTO Nama_tabel [(nama_kolom1,…)]
values (nilai atribut1, …)
Contoh:Masukan data Mahasiswa dengan Nim 10296832, Nama Nurhayati beralamat di Jakarta

INSERT INTO Mahasiswa (Nim, Nama, Alamat) values ("10296832","Nurhayati","Jakarta");

B. **UPDATE**

Sintaks SQL yang digunakan untuk mengubah nilai atribut pada suatu record dari sebuah tabel.
Sintaks : UPDATE nama_tabel
SET nama_kolom = value_1
WHERE kondisi ;
Contoh:
Ubah alamat menjadi "Depok" untuk mahasiswa yang memiliki NIM "10296832"
UPDATE Mahasiswa
SET ALAMAT="Depok"
WHERE NIM=" 10296832";

C. DELETE

Sintaks SQL yang digunakan untuk menghapus record dari sebuah tabel.
Sintaks: DELETE FROM nama_table
WHERE kondisi
Contoh:
Hapus data Mahasiswa yang mempunyai NIM "21198002"
DELETE FROM Mahasiswa
WHERE NIM=" 21198002"

Tabel dibawah ini untuk mengerjakan perintah **SELECT**

Tabel Nilai

NIM	KD_MK	MID	FINAL
10296832	KK021	60	75
10296126	KD132	70	90
31296500	KK021	55	40
41296525	KU122	90	80
21196353	KU122	75	75
50095487	KD132	80	0

Tabel Mahasiswa

NIM	NAMA	ALAMAT
10296832	Nurhayati	Jakarta
10296126	Astuti	Jakarta
31296500	Budi	Depok
41296525	Prananigrum	Bogor
50096487	Pipit	Bekasi
21196353	Quraish	Bogor
10296001	Fintri	Depok
21198002	Julizar	Jakarta

Tabel MataKuliah

KD_MK	NAMA_MK	SKS
KK021	Sistem Basis Data	2
KD132	Sistem Informasi Manajemen	3
KU122	Pancasila	2

D. SELECT

Sintaks : SELECT [DISTINCT | ALL] nama_kolom
FROM nama_tabel
[WHERE condition]
[GROUP BY column_list]
[HAVING condition]
[ORDER BY column_list [ASC | DESC]]

Contoh :

a. Tampilkan semua data Mahasiswa

SELECT NIM,NAMA,ALAMAT FROM Mahasiswa;

Atau

SELECT * FROM Mahasiswa;

Maka hasilnya adalah :

NIM	NAMA	ALAMAT
10296832	Nurhayati	Jakarta
10296126	Astuti	Jakarta
31296500	Budi	Depok
41296525	Prananingrum	Bogor

b. Tampilkan Mata Kuliah yang SKS nya 2

SELECT NAMA_MK FROM MataKuliah WHERE SKS=2

Maka Hasilnya:

NAMA_MK
Sistem Basis Data
Pancasila

c. Tampilkan semua data nilai dimana nilai MID lebih besar sama dengan 60 atau nilai finalnya lebih besar 75.

maka penulisannya :

```
SELECT * FROM Nilai WHERE MID >= 60 OR FINAL > 75
```

Hasilnya:

NIM	KD_MK	MID	FINAL
10296832	KK021	60	75
10296126	KD132	70	90
41296525	KU122	90	80
21196353	KU122	75	75

Aplikasi yang digunakan sebagai contoh adalah Xampp

Dari Address ketik : http://localhost/phpmyadmin

Tampilan user ketik **root** dan **password** dikosongkan

JOIN

JOIN digunakan untuk memilih data dari dua tabel atau lebih.

1. INNER JOIN
   Menggabungkan dua tabel dimana diantara dua tabel datanya bersesuaian.
2. LEFT JOIN atau LEFT OUTER JOIN
   Menggabungkan dua tabel dimana diantara dua tabel datanya bersesuaian dan juga semua record pada tabel sebelah kiri.
3. RIGHT JOIN atau RIGHT OUTER JOIN
   Menggabungkan dua tabel dimana diantara dua tabel datanya bersesuaian dan juga semua record pada tabel sebelah kanan.

Contoh INNER JOIN

```
SELECT Nilai.NIM, Mahasiswa.NAMA, Nilai.KD_MK, Nilai.MID
        FROM Nilai INNER JOIN Mahasiswa
        ON Nilai.NIM = Mahasiswa.NIM
```

Hasil :

NIM	NAMA	KD_MK	MID
10296832	Nurhayati	KK021	60
10296126	Astuti	KD132	70
31296500	Budi	KK021	55
41296525	Prananigrum	KU122	90
21196353	Quraish	KU122	75
50095487	Pipit	KD132	80

Contoh LEFT JOIN

```
SELECT Mahasiswa.NIM, Mahasiswa.NAMA, Nilai.KD_MK, Nilai.MID
        FROM Mahasiswa LEFT OUTER JOIN Nilai
        ON Nilai.NIM = Mahasiswa.NIM
```

Hasil:

NIM	NAMA	KD_MK	MID
10296832	Nurhayati	KK021	60
10296126	Astuti	KD132	70
31296500	Budi	KK021	55
41296525	Prananigrum	KU122	90
21196353	Quraish	KU122	75
50095487	Pipit	KD132	80
10296001	Fintri	-	-
21198002	Julizar	-	-

Contoh RIGHT JOIN

SELECT Mahasiswa.NIM, Mahasiswa.NAMA, Nilai.KD_MK, Nilai.MID
FROM Nilai RIGHT OUTER JOIN Mahasiswa
ON Nilai.NIM = Mahasiswa.NIM

Hasil :

NIM	NAMA	KD_MK	MID
10296832	Nurhayati	KK021	60
10296126	Astuti	KD132	70
31296500	Budi	KK021	55
41296525	Prananigrum	KU122	90
21196353	Quraish	KU122	75
50095487	Pipit	KD132	80
10296001	Fintri	-	-
21198002	Julizar	-	-

Data Access

1. GRANT

Sintaks : GRANT hak_akses ON nama_db
TO nama_pemakai
[IDENTIFIED BY] [PASSWORD] 'Password'
[WITH GRANT OPTION] ;

GRANT hak_akses ON [nama_db]nama_tabel
TO nama_pemakai
[IDENTIFIED BY] [PASSWORD] 'Password'
[WITH GRANT OPTION];

Contoh :
Berikan hak akses kepada Adi untuk menampikan
nilai final test pada tabel Nilai.

GRANT SELECT (FINAL) ON NILAI TO ADI

2. REVOKE

Sintaks : REVOKE hak_akses ON nama_db
FROM nama_pemakai ;

REVOKE hak_akses ON nama_tabel
FROM nama_pemakai ;

Contoh :
Tarik kembali dari Adi hak akses untuk menampilkan nilai final test

REVOKE SELECT (FINAL) ON NILAI FROM ADI

Data Integrity

RECOVER TABLE

Sintaks : RECOVER TABLE nama_tabel

Contoh :

Kembalikan keadaan data mahasiswa seperti pada saat sebelum terjadi kerusakan

RECOVER TABLE MHS ;

Auxiliary

1. **SELECT … INTO OUTFILE** 'filename'

Sintaks ini digunakan untuk mengekspor data dari tabel ke file lain.

Sintaks :
```
SELECT … INTO
OUTFILE 'Nama File'
        [FIELDS | COLUMNS]
        ['TERMINATED BY 'string']
[[OPTIONALLY] ENCLOSED BY 'char']
        [ESCAPED BY 'char'] ]
```

Contoh :

Ubah semua data mahasiswa ke bentuk ASCII dan disimpan ke file teks di directory/home/adi dengan pemisah antar kolom '|'

```
SELECT * FROM MHS
INTO OUTFILE "/home/adi/teks"
FIELDS TERMINATED BY " |";
```

2. LOAD

Sintaks query ini digunakan untuk mengimpor data dari file lain ke tabel.

Sintaks :
```
LOAD DATA INFILE " nama_path"
INTO TABLE nama_tabel [ nama_kolom] ;
[FIELDS | COLUMNS]
['TERMINATED BY 'string']
[[OPTIONALLY] ENCLOSED BY 'char']
[ESCAPED BY 'char'] ]
```

Contoh :

Memasukkan data-data dari file teks yang berada pada direktori "/home/adi" ke dalam tabel MHS_2. Dimana pemisah antara kolom dalam file teks adalah tab (\t) :

```
LOAD FROM "/home/adi/teks"
INTO MHS_2
FILELDS TERMINATED BY '\t';
```

3. RENAME TABLE

Sintaks :

```
RENAME TABLE  OldnamaTabel
TO NewNamaTabel
```

Contoh :

```
RENAME TABLE MHS
TO MAHASISWA
```

Fungsi Aggregate

Menggunakan Fungsi Aggregate :

1. COUNT digunakan untuk menghitung jumlah.
   Menghitung jumlah record mahasiswa dari tabel MAHASISWA
   SELECT COUNT(*) FROM MAHASISWA
2. SUM digunakan untuk menghitung total dari kolom yang mempunyai tipe data numerik.
   SELECT SUM(SKS) AS 'TOTAL SKS' FROM MATAKULIAH
3. AVG digunakan untuk menghitung rata-rata dari data-data dalam sebuah kolom.
   SELECT AVG(FINAL) AS 'FINAL' FROM Nilai
4. MIN digunakan untuk menghitung nilai minimal dalam sebuah kolom.
   SELECT MIN(FINAL) FROM Nilai
5. MAX diguankan untuk menghitung nilai maksimum dalam sebuah kolom
   SELECT MAX(MID) FROM Nilai

Subquery

SUBQUERY

Adalah subselect yang dapat digunakan di klausa WHERE dan HAVING dipernyataan select luar untuk menghasilkan tabel akhir.

Aturan-aturan untuk membuat subquery, yaitu :

1. Klausa Order By tidak boleh digunakan di subquery, Order By hanya dapat digunakan di pernyataan Select luar.
2. Klausa subquery Select harus berisi satu nama kolom tunggal atau ekspresi kecuali untuk subquery-subquery menggunakan kata kunci EXIST
3. Secara default nama kolom di subquery mengacu ke nama tabel di klausa FROm dari subquery tersebut.
4. Saat subquery adalah salah satu dua operan dilibatkan di pembandingan, subquery harus muncul disisi kanan pembandingan

Penggunanaan ANY dan ALL

Jika subquery diawali kata kunci ALL, syarat hanya akan bernilai TRUE jika dipenuhi semua nilai yang dihasilkan subquery itu.

Jika subquery diawali kata kunci ANY, syaratnya akan bernilai TRUE jika dipenuhi sedikitnya satu nilai yang dihasilkan subquery tersebut.

Penggunanaan EXIST DAN NOT EXIST

EXIST akan mengirim nilai TRUE jika dan hanya jika terdapat sedikitnya satu baris di tabel hasil yang dikirim oleh subquery dan EXIST mengirim nilai FALSE jika subquery mengirim tabel kosong. Untuk NOT EXIST kebalikan dari EXIST. (Masing-masing dosen membuat contoh untuk subquery)

CONTOH SUBQUERY :

1. Ambil nilai mid dan final dari mahasiswa yang bernama Astuti.
   SELECT MID, FINAL FROM NILAI WHERE NIM=(SELECT NIM FROM MAHASISWA WHERE NAMA='Astuti')
2. Ambil nilai kode matakuliah, mid dan final dari mahasiswa yang tinggal di jakarta.
   SELECT KD_MK, MID, FINAL FROM NILAI WHERE NIM IN(SELECT NIM FROM MAHASISWA WHERE ALAMAT = 'Jakarta')
3. Ambil nama-nama mahasiswa yang mengikuti ujian.
   SELECT NAMA FROM MAHASISWA WHERE EXISTS (SELECT NIM FROM NILAI WHERE NILAI.NIM= MAHASISWA.NIM)
4. Ambil nama-nama mahasiswa yang tidak mengikuti ujian.
   SELECT NAMA FROM MAHASISWA WHERE NOT EXISTS (SELECT NIM FROM NILAI WHERE NILAI.NIM= MAHASISWA.NIM)

BAB IX
Basis Bata Terdistribusi

Basis Data Terdistribusi

Yaitu kumpulan data yang digunakan bersama yang saling terhubung secara logik tetapi tersebar secara fisik pada suatu jaringan komputer.

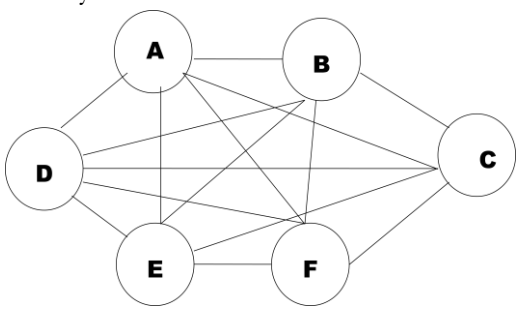
Karakteristik Database terdistribusi, yaitu :

1. Kumpulan data yang digunakan bersama secara logik tersebar pada sejumlah komputer yang berbeda
2. Komputer yang dihubungkan menggunakan jaringan komunikasi
3. Data pada masing-masing situs dapat menangani aplikasi-aplikasi lokal secara otonom
4. Data pada masing situs dibawah kendali satu DBMS
5. Masing-masing DBMS berpartisipasi dalam sedikitnya satu aplikasi global

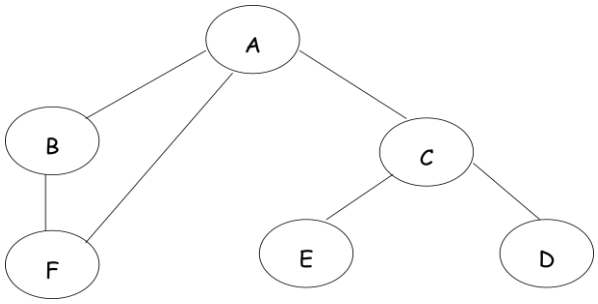
Topologi Distribusi Data

BENTUK-BENTUK TOPOLOGI DISTRIBUSI DATA :

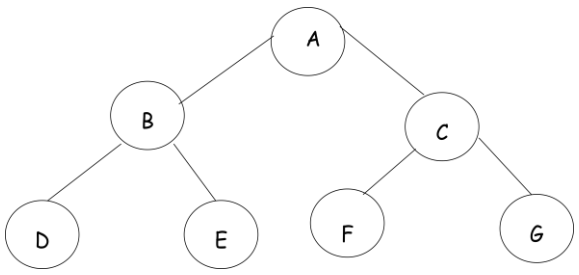
a. Fully Connected network



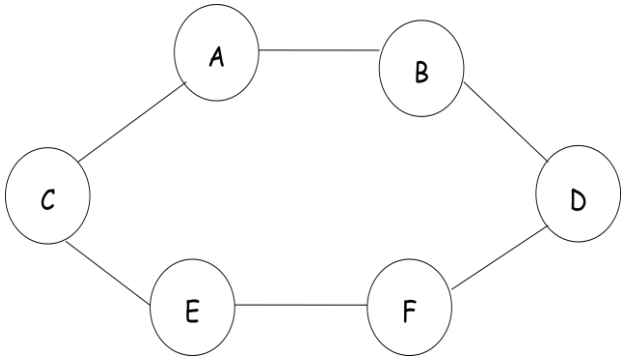
b. Partialy conneted network



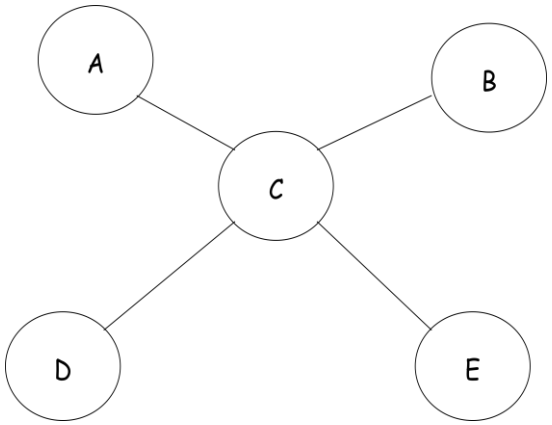
C. Tree Strutured Network



d. Ring network



e. Star network



Keuntungan Basis Data Terdistribusi:

1. Secara alami mengikuti struktur organisasi
2. Adanya otonomi lokal
3. Sifatnya dapat dipakai secara bersama
4. Peningkatan ketersediaan
5. Peningkatan kehandalan
6. Peningkatan kinerja
7. Ekonomis
8. Pertumbuhan yang modular

Kerugian Basis Data Terdistribusi:

1. Harga software mahal (Biaya)
2. Kompleksitas
3. Kelemahan dalam keamanan
4. Sulitnya menjaga keutuhan data
5. Kurangnya standar
6. Kurangnya pengalaman
7. Perancangan basisdata lebih kompleks

Fragmentasi Data

FRAGMENTASI Merupakan sebuah proses pembagian atau pemetaan database dimana database dipecah-pecah berdasarkan kolom dan baris yang kemudian disimpan didalam site

atau unit komputer yang berbeda dalam suatu jaringan data, sehingga memungkinkan untuk pengambilan keputusan terhadap data yang telah terbagi.

Fragmentasi data merupakan langkah yang diambil untuk menyebarkan data dalam basis data terdistribusi.

Alasan-alasan diperlukannya fragmentasi, yaitu :

1. Penggunaan
2. Efisiensi
3. Paralleslisme
4. Keamanan

Beberapa Peraturan Yang Harus Didefinisikan Ketika Mendefinisikan Fragment :

1. Kondisi lengkap (*Completeness*)
   sebuah unit data yang masih dalam bagian dari relasi utama, maka data harus berada dalam satu fragmen. Ketika ada relasi, pembagian datanya harus menjadi satu kesatuan dengan relasinya.
2. Rekontruksi (*Reconstruction*)
   sebuah relasi asli dapat dibuat kembali atau digabungkan kembali dari sebuah fragmen. Ketika telah dipecah-pecah, data masih memungkinkan untuk digabungkan kembali dengan tidak mengubah struktur data.
3. Disjointness
   data didalam fragmen tidak boleh diikutkan dalam fragmen lain agar tidak terjadi redundancy data, kecuali untuk atribut primary key dalam fragmentasi vertikal

Kerugian fragmentasi yaitu :

1. Kinerja yang dapat turun karena data tersebar dan butuh proses untuk penggabungan kembali
2. Integritas yang dapat terganggu dikarenakan kegagalan pada salah satu site database server

TIGA JENIS FRAGMENTASI :

1. **Fragmentasi horizontal**
   terdiri dari tuple dari fragment global yang kemudian dipecah-pecah atau disekat menjadi beberapa sub-sets
2. **Fragmentasi vertikal**
   Membagi atribut-atribut dari fragment global yang tersedia menjadi beberapa grup.
3. **Fragmentasi campuran**
   Cara yang sederhana untuk membangun fragmentasi campuran sbb :
   a. Menggunakan fragmentasi horizontal pada fragmentasi vertikal
   b. Menggunakan fragmentasi vertical pada fragmentasi horizontal

CONTOH KASUS JENIS-JENIS FRAGMENTASI

Ujian (NIM,Nama_Mhs,Kode_MK,Mt_Kuliah,Nil_Akhir,Grade)

NIM	Nama_Mhs	Kode_MK	Mt_Kuliah	Nil_Akhir	Grade
123	Fathi	101	Sistem Basis Data	78	B
124	Farah	102	Peranc. Sistem	60	C
125	Sarah	101	Sistem Basis Data	40	D
126	Salsabila	101	Sistem Basis Data	90	A
127	Azizah	103	Visual Basic	70	B
128	Farhan	103	Visual Basic	40	D
129	Faiz	102	Peranc. Sistem	80	A

Contoh Fragmentasi Horizontal

Fragmentasi **Horizontal** terbagi menjadi 3 fragment yang berbeda berdasarkan Mt_Kuliah

1. Relasi Mt_Kuliah="Sistem Basis Data"

   σ Mt_Kuliah="Sistem Basis Data" (Ujian)

NIM	Nama_Mhs	Kode_MK	Mt_Kuliah	Nil_Akhir	Grade
123	Fathi	101	Sistem Basis Data	78	B
125	Sarah	101	Sistem Basis Data	40	D
126	Salsabila	101	Sistem Basis Data	90	A

2. Relasi Mt_Kuliah="Peranc. Sistem"

   σ Mt_Kuliah="Peranc. Sistem" (Ujian)

NIM	Nama_Mhs	Kode_MK	Mt_Kuliah	Nil_Akhir	Grade
124	Farah	102	Peranc. Sistem	60	C
129	Faiz	102	Peranc. Sistem	80	A

3. Relasi Mt_Kuliah="Visual Basic"

   σ Mt_Kuliah="Visual Basic" (Ujian)

NIM	Nama_Mhs	Kode_MK	Mt_Kuliah	Nil_Akhir	Grade
127	Azizah	103	Visual Basic	70	B
128	Farhan	103	Visual Basic	40	D

Fragment di atas memenuhi kondisi jika Nama_Mhs dan Mt_Kuliah adalah hal-hal yang memenuhi syarat **Fragmentasi vertical**: berdasarkan dekomposisi-nya dengan menambahkan tupel_id

NIM	Nama_Mhs	Kode_MK	Mt_Kuliah	Nil_Akhir	Grade	Tuple_ID
123	Fathi	101	Sistem Basis	78	B	1
124	Farah	102	Data	60	C	2
125	Sarah	101	Peranc. Sistem	40	D	3
126	Salsabila	101	Sistem Basis	90	A	4
127	Azizah	103	Data	70	B	5
128	Farhan	103	Sistem Basis	40	D	6
129	Faiz	102	Data	80	A	7
			Visual Basic			
			Visual Basic			
			Peranc. Sistem			

Relasi 1 = NIM, Nama_Mhs, Mt,Kuliah, Nil_Akhir, Grade, Tuple_ID

π NIM,Nama_Mhs,Mt,Kuliah,Nil_Akhir,Grade,Tuple_ID (UJian)

NIM	Nama_Mhs	Mt_Kuliah	Nil_Akhir	Grade	Tuple_ID
123	Fathi	Sistem Basis Data	78	B	1
124	Farah	Peranc. Sistem	60	C	2
125	Sarah	Sistem Basis Data	40	D	3
126	Salsabila	Sistem Basis Data	90	A	4
127	Azizah	Visual Basic	70	B	5
128	Farhan	Visual Basic	40	D	6
129	Faiz	Peranc. Sistem	80	A	7

Relasi 2 = NIM,Kode_MK,Nil_Akhir,Grade,Tuple_ID

π NIM,Kode_MK,Nil_Akhir,Grade,Tuple_ID (Ujian)

NIM	Kode_MK	Nil_Akhir	Grade	Tuple_ID
123	101	78	B	1
124	102	60	C	2
125	101	40	D	3
126	101	90	A	4
127	103	70	B	5
128	103	40	D	6
129	102	80	A	7

Terdapat relasi berdasarkan Mata Kuliah yang sama

Relasi 1a.

π NIM,Nama_Mhs,Mt_Kuliah,Nil_Akhir,Grade,Tuple_ID(σ Mt_Kuliah="Sistem Basis Data" (Ujian))

NIM	Nama_Mhs	Mt_Kuliah	Nil_Akhir	Grade	Tuple_ID
123	Fathi	Sistem Basis Data	78	B	1
125	Sarah	Sistem Basis Data	40	D	3
126	Salsabila	Sistem Basis Data	90	A	4

Relasi 1b.

π NIM,Nama_Mhs,Mt_Kuliah,Nil_Akhir,Grade,Tuple_ID(σ Mt_Kuliah="Peranc. Sistem" (Ujian))

NIM	Nama_Mhs	Mt_Kuliah	Nil_Akhir	Grade	Tuple_ID
124	Farah	Peranc. Sistem	60	C	2
129	Faiz	Peranc. Sistem	80	A	7

Relasi 1c

π NIM,Nama_Mhs,Mt_Kuliah,Nil_Akhir,Grade,Tuple_ID(σ Mt_Kuliah="Visual Basic" (Ujian))

NIM	Nama_Mhs	Mt_Kuliah	Nil_Akhir	Grade	Tuple_ID
127	Azizah	Visual Basic	70	B	5
128	Farhan	Visual Basic	40	D	6

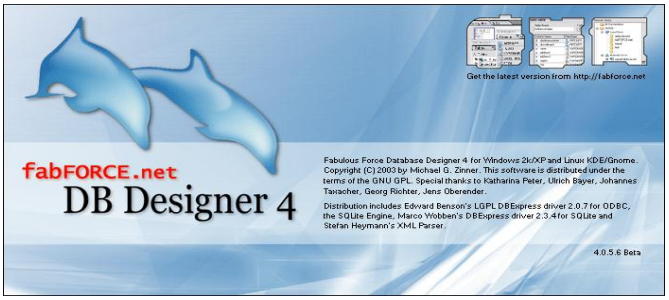
BAB X
Perancangan dan Implementasi Basis Data Menggunakan DB Designer

10.1 Perancangan dan Implementasi Basis Data Menggunakan MYSQL

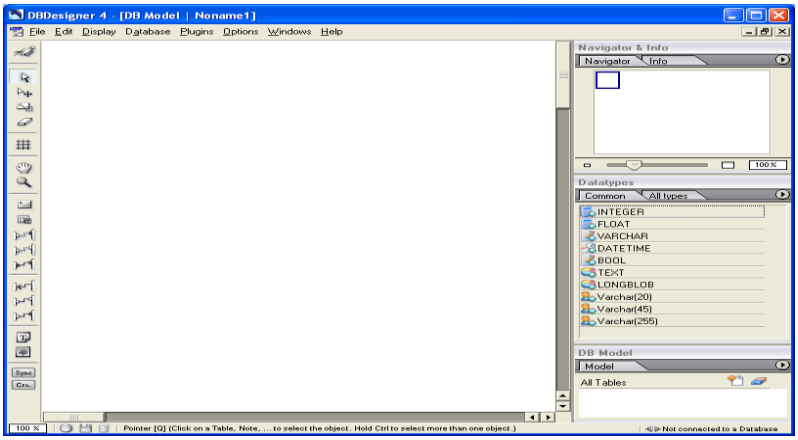
Perangkat Lunak Bantu untuk Perancangan Basis Data

Pada perangkat lunak bantu telah tersedia komponen-komponen (notasi-notasi) perancangan basis data.

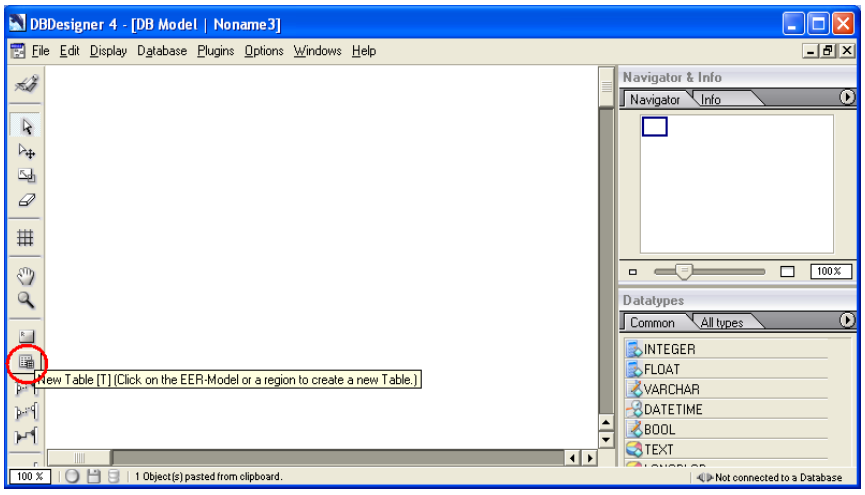
Salah satu perangkat lunak bantu untuk keperluan semacam itu adalah DBDesigner yang dioptimalkan untuk MySQL Database.



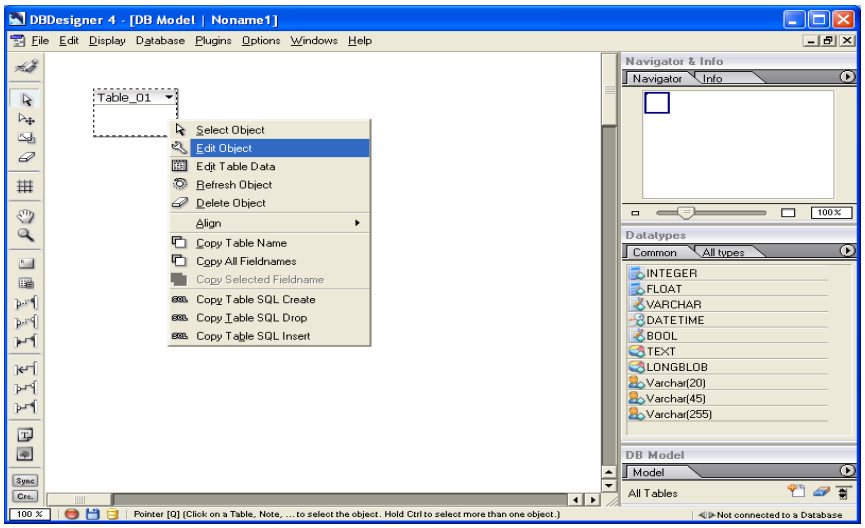
- Tampilan jendela DBDesigner.



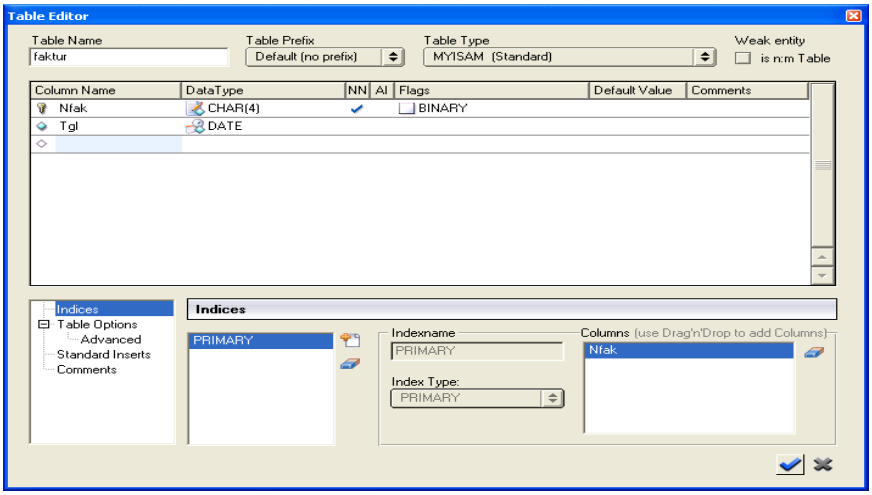
Menggunakan Komponen TABEL dan RELASI Klik komponen **Tabel** pada toolbar seperti di gambar berikut.



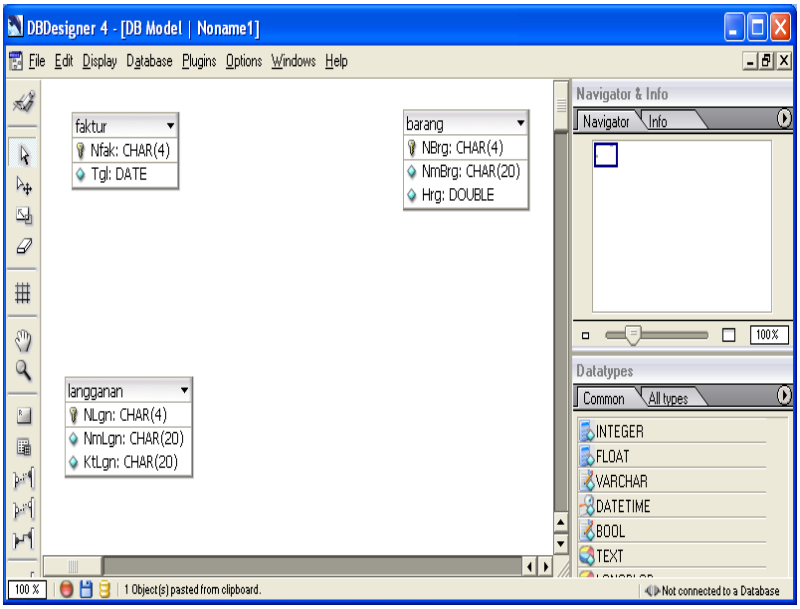
Letakan komponen tsb. pada page area sehingga muncul komponen **Tabel** (Table_01) pada page area, kemudian klik kanan komponen tsb sehingga muncul menu dan pilihlah **Edit Object** seperti berikut.



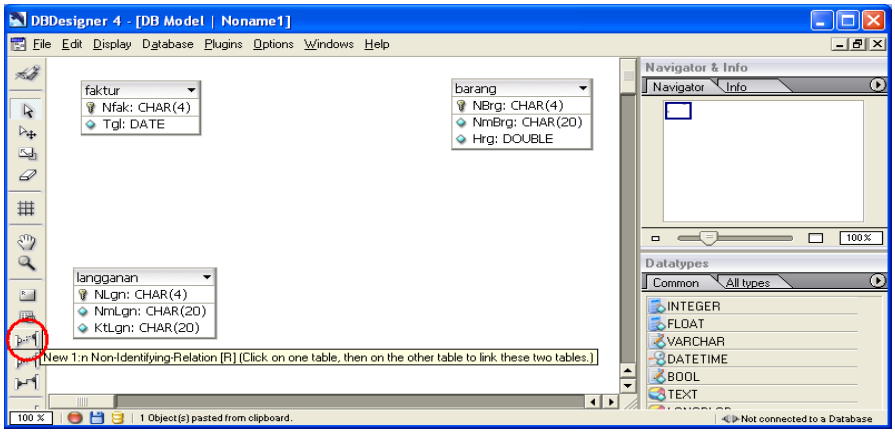
Menu Edit Object akan menampilkan jendela **Table Editor**.
Pada **Table Editor** kita bisa menentukan properties dari tabel seperti nama tabel, tipe data, primary key dsb.
Ubah dan simpanlah properties tabel (Table _01) menjadi tabel **faktur** (struktur tabel seperti pada pembahasan LRS tanpa ada FK) seperti berikut.



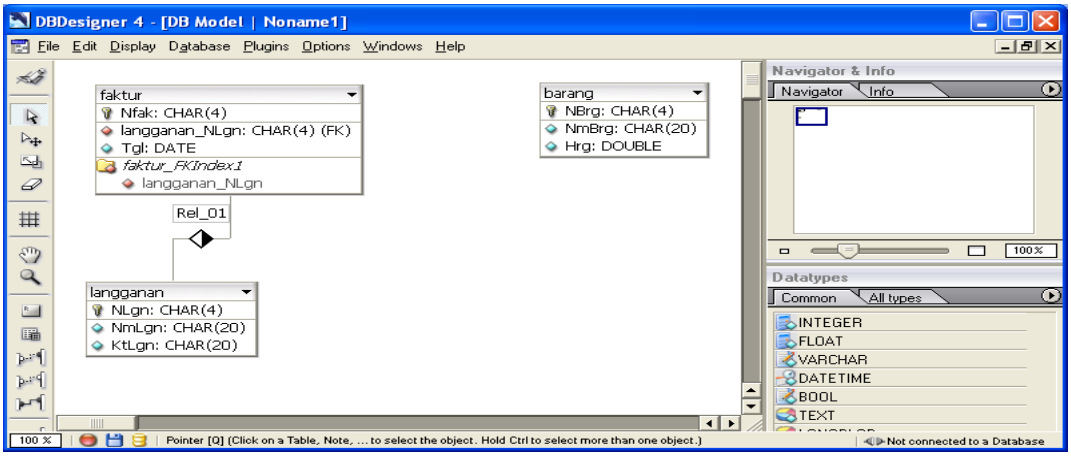
Ulangi langkah-langkah menggunakan komponen **Table** di atas (**tabel faktur**) untuk tabel **barang** dan **langganan** (struktur tabel seperti pada pembahasan LRS tanpa ada FK). Sehingga ada 3 komponen Table seperti gambar berikut



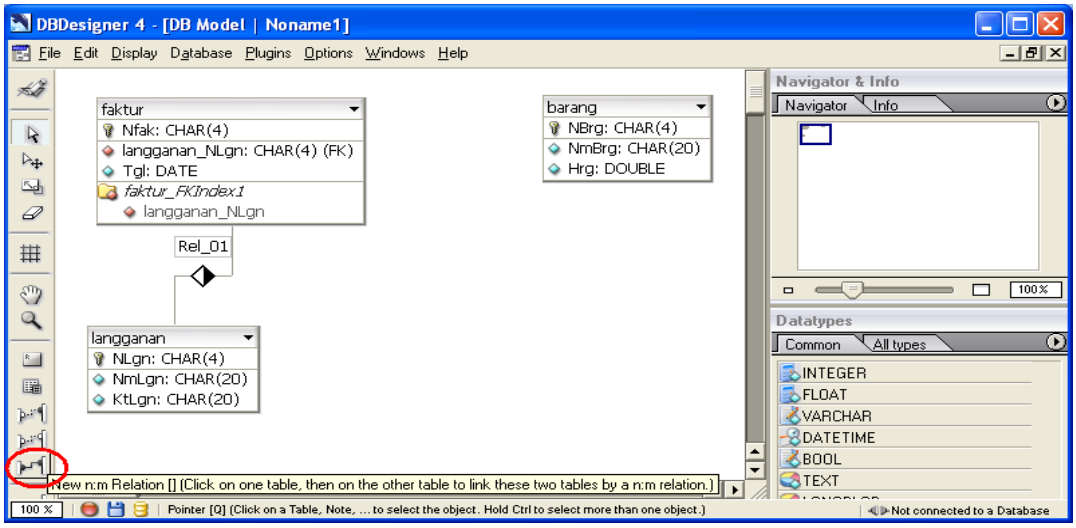
Langkah berikutnya membuat relasi **1-M** antara **langganan** dengan **faktur** dengan cara klik komponen **1-n Relation** pada toolbar seperti di gambar berikut.



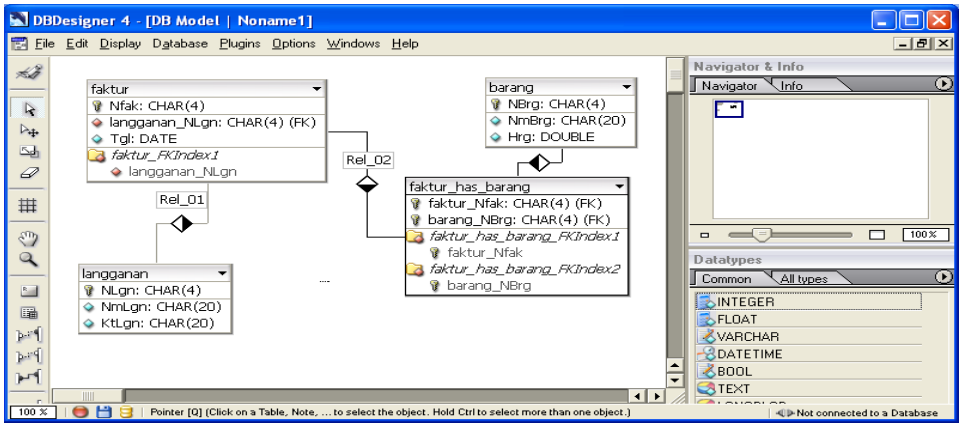
Klik di tabel **langganan** kemudian klik di tabel **faktur**, sehingga muncul komponen relasi yang menghubungkan kedua tabel tsb. dan FK (NLgn) berada pada tabel faktur, seperti gambar berikut



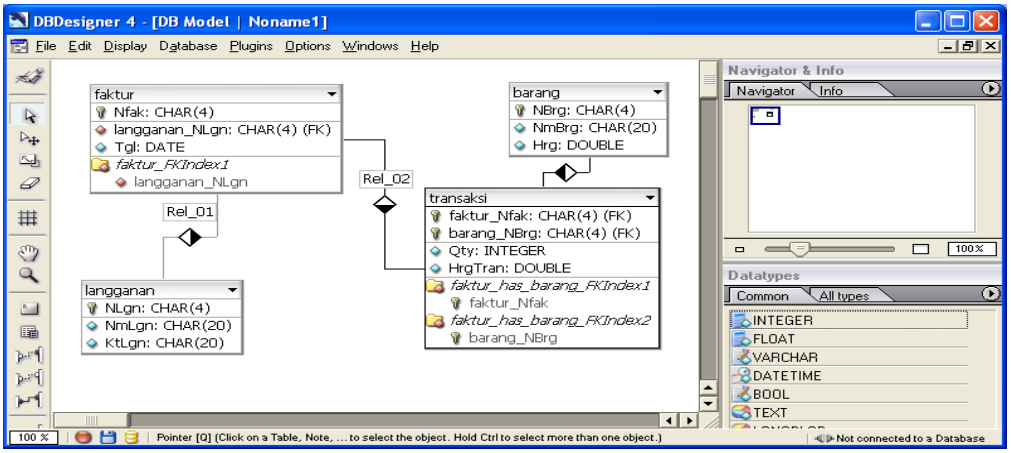
Langkah berikutnya membuat relasi **M-M** antara **faktur** dengan **barang** dengan cara klik komponen **n-m Relation** pada toolbar seperti di gambar berikut



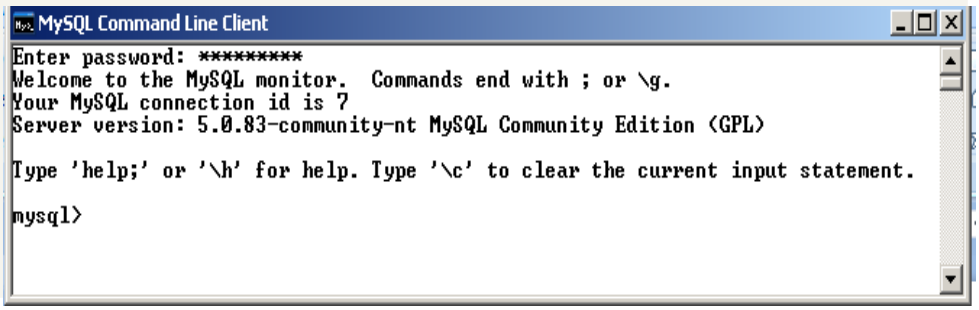
Klik di tabel **faktur** kemudian klik di tabel **barang**, sehingga muncul komponen relasi yang disertai munculnya tabel baru (**faktur_has_barang**) dan FK (Nfak & NBrg) berada pada tabel tsb, seperti gambar berikut.



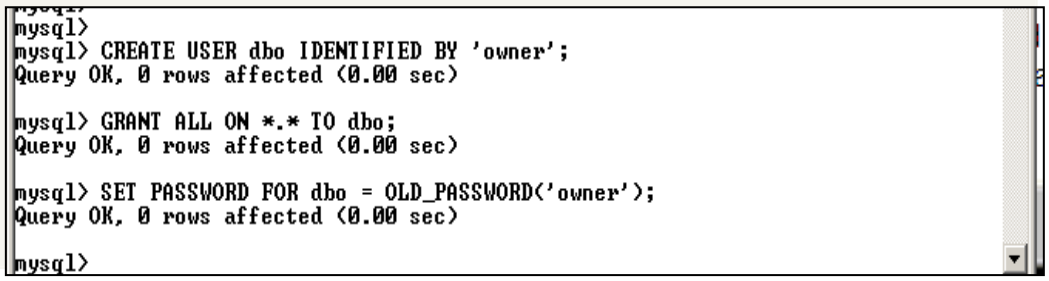
Edit properties tabel faktur_has_barang yaitu dengan mengganti nama menjadi tabel transaksi dan menambahkan field Qty dan HrgTran. Sehingga menjadi seperti gambar berikut.



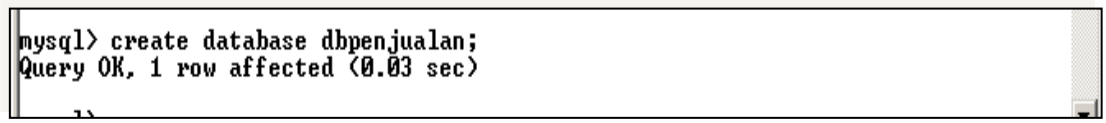
Untuk mengekspor hasil rancangan database ke dalam database digunakan **Database Synchronization**. Database yang digunakan pada contoh ini adalah MySQL.
Sebelum melakukan sinkronisasi, kita perlu membuat koneksi ke database MySQL terlebih dahulu. Jika remote connection dengan root diperbolehkan maka gunakan user root. Jika tidak maka kita butuh membuat user baru terlebih dahulu. Berikut ini adalah cara bagaimana membuat user baru yaitu db_owner.
Lakukan login terlebih dahulu ke MySQL dengan memasukkan password root.



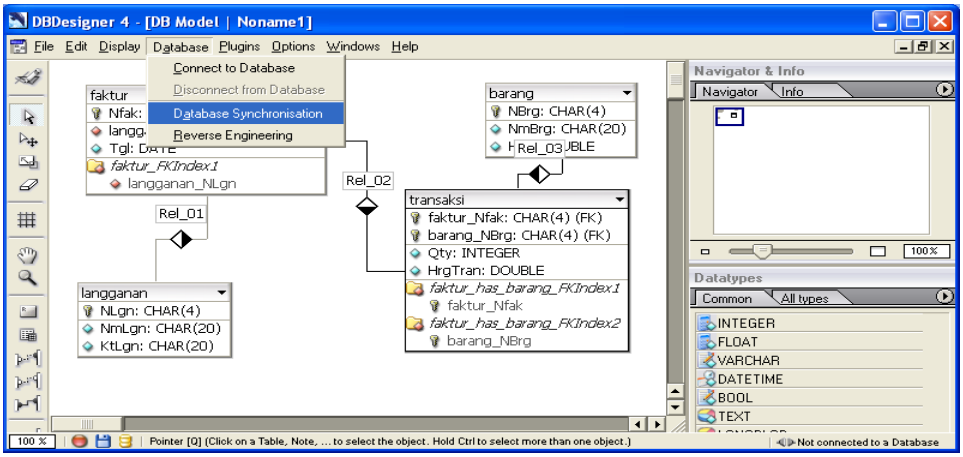
Buat user baru bernama dbo dengan password ”owner”. Ketikkan 3 perintah dibawah ini.



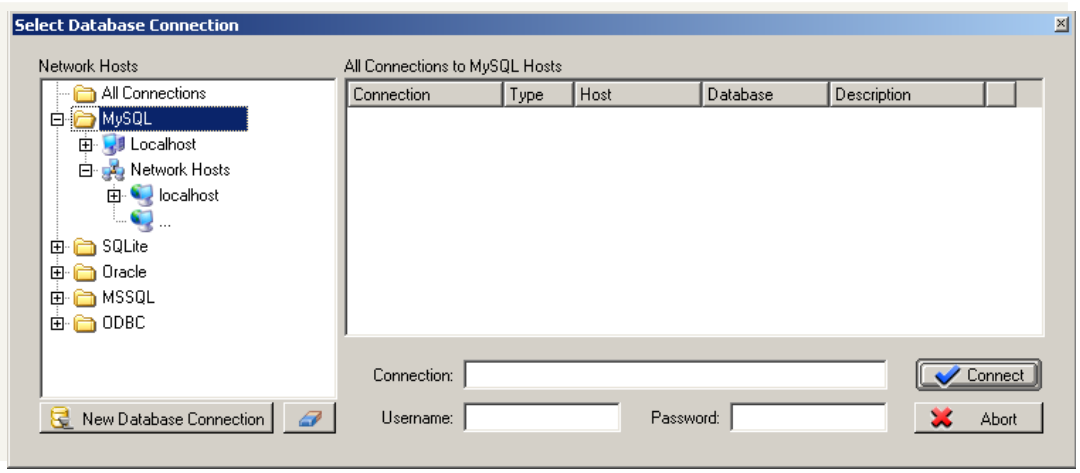
Buat Database baru yaitu dbpenjualan

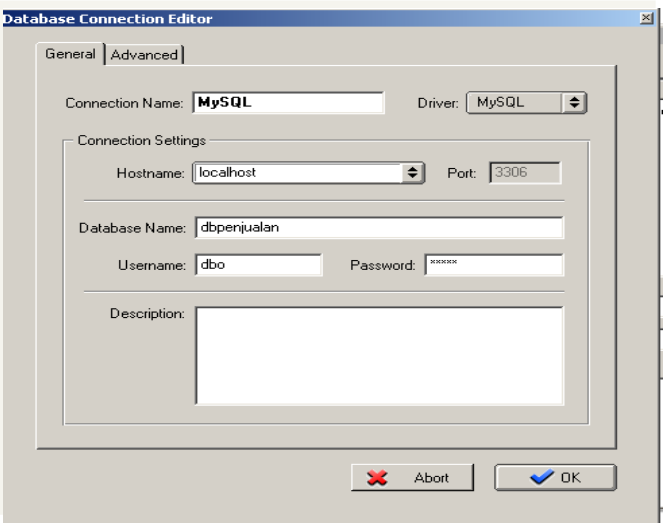


Mengekspor Tabel Hasil Rancangan Ke Server Database
Mengekspor tabel ke server database bisa dilakukan dari menu
Database → Database Sychronisastion seperti gambar berikut.



Lalu pilih MySQL sebagai database dan kemudian klik **New Database Connection**





Masukkan Nilai berikut:

Connection Name : *MySQL*

Hostname : *localhost*

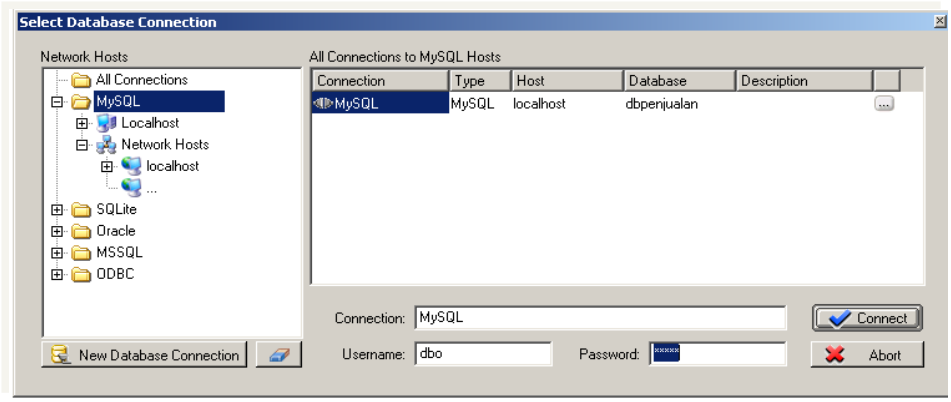
Database Name : *dbpenjualan*

UserName : *dbo*

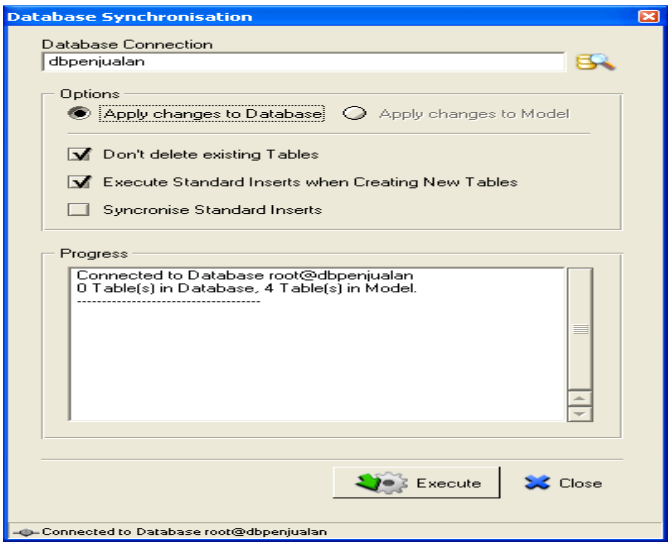
Password : *owner*

Lalu klik OK

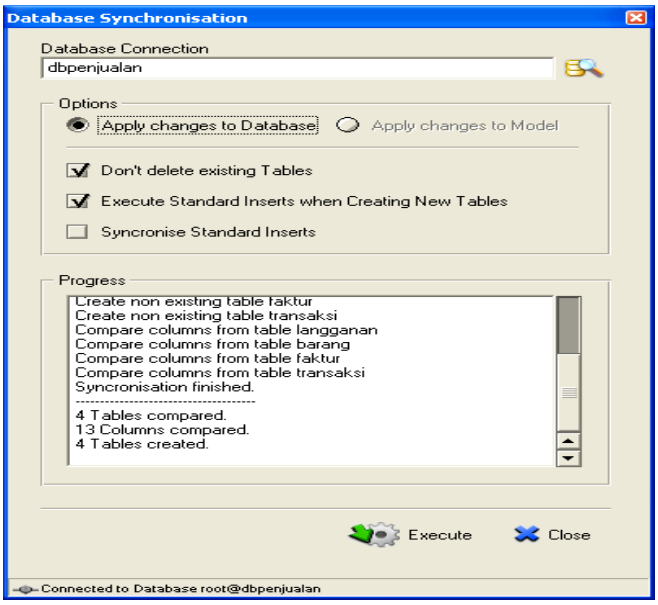
Klik **Connect** untuk terkoneksi ke MySQL



Klik **Execute** untuk mengeksekusi sinkronisasi



Setelah tampil jendela seperti di atas, selanjutnya klik tombol **EXECUTE** untuk mengekspor tabel ke server database MySQL dan akan tampil progress report seperti berikut



Latihan 1

1. Sebuah perusahaan yang melayani pemesanan barang/produk umum memerlukan sebuah program aplikasi yang berfungsi untuk menyimpan data produk beserta suppliernya dan juga berfungsi untuk mencatat transaksi pemesananan produk dari customer. Setiap produk yang dipesan akan dikirim ke customer yang memesannya. Rancanglah database untuk program aplikasi tersebut dengan menggunakan DBdesigner dan ekspor hasilnya ke server MySQL, untuk memenuhi keinginan perusahaantersebut.
2. Seorang kolektor mobil ingin mendata seluruh mobil miliknya dan memerlukan program aplikasi yang bisa berfungsi untuk menyimpan data koleksi mobilnya. Rancanglah database untuk program aplikasi tersebut dengan menggunakan DBdesigner dan ekspor hasilnya ke server MySQL, sehingga program yang dikembangkan bisa memenuhi keinginan kolektor tersebut.

BAB XI
Lingkungan Basis Bata

11.1 KONKURENSI

CONCURRENCY (KONKURENSI)

Ada 3 masalah yang disebabkan oleh Concurrency :

1. Masalah kehilangan modifikasi (*Lost Update Problem*)

   Masalah ini timbul jika dua transaksi mengakses item database yang sama yang mengakibatkan nilai dari database tersebut menjadi tidak benar.

Transaksi A	Waktu	Transaksi B
= Baca R = = = Modifikasi R = = =	↓ t1 ↓ t2 ↓ t3 ↓ T4 ↓	= = = Baca R = = = Modifikasi R =

Contoh Lost Update problem

Data transaksi pada rekening bersama (Ika dan Susi)

Waktu	Transaksi Ika	Transaksi Susi	Saldo
T1	Read Saldo	……	1.000.000
T2	……	Read Saldo	1.000.000
T3	Saldo:=Saldo-50.000	……..	1.000.000
T4	Write Saldo	……..	950.000
T5	……	Saldo:= saldo+100.000	1.000.000
T6	……	Write Saldo	1.100.000

Nilai saldo menjadi tidak benar disebabkan transaksi Susi membaca nilai saldo sebelum transaksi Ika mengubah nilai tersebut dalam database, sehingga nilai yang sudah di update yang dihasilkancdari transaksi Ika menjadi hilang.

2. Masalah Modifikasi Sementara (*uncommited Update Problem*)

   Masalah ini timbul jika transaksi membaca suatu record yang sudah dimodifikasi oleh transaksi lain tetapi belum terselesaikan (*uncommited*), terdapat kemungkinan kalau transaksi tersebut dibatalkan (*rollback*).

Transaksi A	Waktu	Transaksi B
- Baca R - - - Modifikasi R -	↓ t1 ↓ t2 ↓ t3 ↓	- Modifkasi R - - - Rollback -

Contoh *uncommited Update Problem*

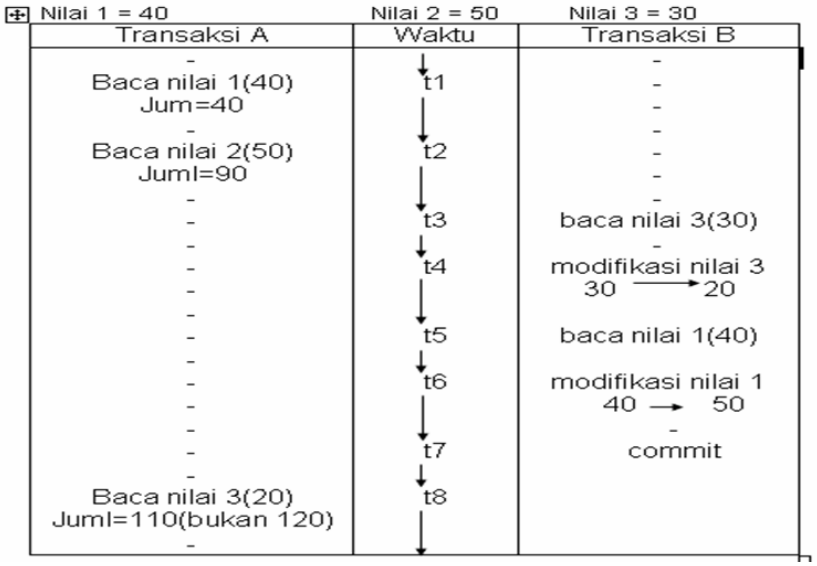
Waktu	Transaksi Simpanan	Transaksi Bunga	Saldo
T1	Read Saldo	.....	1.000.000
T2	Saldo:=saldo+1.000.0000	.....	1.000.000
T3	Write Saldo	.....	2.000.000
T4	.....	Read Saldo	2.000.000
T5	.....	Saldo:= saldo*0.15	2.000.000
T6	.....	Write Saldo	2.300.000
T7	.....	RollBack	2.300.000

Nilai saldo menjadi tidak benar disebabkan terjadi RollBack pada T7 yanmembatalkan transaksi sebelumnya (T6), sehingga saldo seharusnya tetap 2.000.000.

3. Masalah Analisa yang tidak konsisten (*Problem of inconsistency Analysis*)

Masalah ini timbul jika sebuah transaksi membaca suatu nilai tetapi transaksi yang kedua mengupdate beberapa nilai tersebut selama eksekusi transaksi pertama.

Contoh Problem of Inconsistency Analysis



Transaksi A menjumlahkan nilai 1, nilai 2 dan nilai 3

Transaksi B → nilai 1 + 10, nilai 3 –10

11.2 LOCKING

LOCKING adalah salah satu mekanisme pengontrol concurrency

KONSEP DASAR :

Ketika sebuah transaksi memerlukan jaminan kalau record yang diinginkan tidak akan berubah secara mendadak, maka diperlukan kunci untuk record tersebut

FUNGSI

Locking berfungsi untuk menjaga record tersebut agar tidak dimodifikasi oleh transaksi lain.

Jenis- Jenis Lock :

1. Share (S)

   Kunci ini memungkinkan pengguna dan para pengguna konkuren yang lain dapat membaca record tetapi tidak mengubahnya.

2. Exclusive (X)

   Kunci ini memungkinkan pengguna untuk membaca dan mengubah record. Sedangkan pengguna konkuren lain tidak diperbolehkan membaca ataupun mengubah record tersebut.

	X	S	-
X	N	N	Y
S	N	Y	Y
-	Y	Y	Y

X = kunci X
S = kunci S
N = No
Y = Yes

Cara Kerja Locking

Masalah kehilangan modifikasi (Lost Update Problem)

Transaksi A	Waktu	Transaksi B
-		-
baca R11 (kunci S)		-
-	t2	baca R(kunci S)
-	t3	-
modifikasi R (kunci X)	t4	-
tunggu		modifikasi R (kunci X)
.		tunggu
.		.
.		.
.		tunggu
.		
tunggu		

Masalah Modifikasi Sementara (uncommited Update Problem)

Transaksi A	Waktu	Transaksi B
-		-
-	t1	modifikasi R (kunci X)
-	t2	-
baca R kunci (S)		-
tunggu	t3	-
.		synchpoint (kunci X dilepas)
.		-
tunggu	t4	-
baca R kembali (Kunci S)		-

Transaksi A	Waktu	Transaksi B
-	t1	modifikasi R
-		(kunci X)
Modifikasi R	t2	-
Kunci (X)		-
tunggu		-
:	t3	synchpoint
:		(kunci X dilepas)
tunggu		-
modifikasi R	t4	-
(Kunci X)		-

Masalah Analisa yang tidak konsisten (Problem of inconsistensi Analisa)

Nilai 1 = 40 Nilai 2 = 50 Nilai 3 = 30

Transaksi A	Waktu	Transaksi B
-		-
Baca nilai 1(40)	t1	-
(kunci S)		-
Jumlah=40		-
-		-
Baca nilai 2(50)	t2	-
(kunci S)		-
Jumlah=90		
-	t3	baca nilai 3(30)
-		(kunci S)
-		-
-	t4	modifikasi nilai 3
-		(kunci X)
-		30 → 20
-		-
-	t5	baca nilai 1(40)
-		(kunci S)
-		-
-	t6	modifikasi nilai 1
-		(kunci X)
-		tunggu
modifikasi nilai 3	t7	:
(kunci S)		:
tunggu		tunggu

11.3 TIMESTAMPING

Adalah salah satu alternatif mekanisme kontrol konkurensi yang dapat menghilangkan masalah dead lock

Dua masalah yang timbul pada Timestamping :

1. Suatu transaksi memerintahkan untuk membaca sebuah item yang sudah di update oleh transaksi yang belakangan.
2. Suatu transaksi memerintahkan untuk menulis sebuah item yan nilainya sudah dibaca atau ditulis oleh transaksi yang belakangan

11.4 CRASS DAN RECOVERY

PENGERTIAN :

Crash adalah suatu failure atau kegagalan dari suatu sistem

PENYEBAB DARI KEGAGALAN ADALAH :

1. Disk Crash yaitu informasi yang ada di disk akan hilang
2. Power failure yaitu informasi yang disimpan pada memori utama dan register akan hilang
3. Software Error yaitu output yang dihasilkan tidak betul dan sistem databasenya sendiri akan memasuki suatu kondisi tidak konsisten

Klasifikasi Failure

Berdasarkan Jenis storage

1. Volatile storage, biasanya informasi yang terdapat pada volatile akan hilang, jika terjadi kerusakan sistem (system crash) contoh: RAM
2. Non Volatile Storage, biasanya informasi yang terdapat pada non volatile strorage tidak akan hilang jika terjadi kerusakan sistem contoh: ROM
3. Stable Storage, informasi yang terdapat dalam stable storage tidak pernah hilang. contoh: Harddisk RAID

Jenis-Jenis Kegagalan

1. Logical Error, program tidak dapat lagi dilaksanakan disebabkan oleh kesalahan input, data tidak ditemukan, over flow
2. System Error, sistem berada pada keadaan yang tidak diinginkan, seperti terjadi deadlock, sebagai akibat program tidak dapat dilanjutkan namun setelah beberapa selang waktu program dapat dijalankan kembali.
3. System Crash,kegagalan fungsi perangkat keras, menyebabkan hilangnya data pada volatile storage, tetapi data pada non volatile storage masih tetap ada.
4. Disk Failure, hilangnya data dari sebuah blok disk disebabkan oleh kerusakan head atau kesalahan pada waktu pengoperasian transfer data

11.5 SECURITY

SECURITY adalah suatu proteksi data terhadap perusakan data dan pemakaian oleh pemakai yang tidak mempunyai ijin.

Beberapa Masalah Security Secara Umum :

1. Di dalam suatu perusahaan siapa yang diijinkan untuk mengakses suatu sistem
2. Bila sistem tersebut menggunakan password, bagaimana kerahasian dari password tersebut dan berapa lama password tersebut harus diganti
3. Di dalam pengontrolan hardware, apakah ada proteksi untuk penyimpanan data (data storage)

Dua Katagori Penyalahgunaan Database :

1. Katagori yang tidak disengaja
   Contoh: Anomali yang disebabkan oleh pendistribusian data pada beberapa komputer
2. Katagori yang disengaja
   Contoh**:** Insert, Delete & Update oleh pihak yang tidak berwenang

Beberapa Tingkatan Masalah Security :

1. Phisical, berkaitan dengan pengamanan lokasi fisik database
2. Man, berkaitan dengan wewenang user
3. Sistem operasi, berkaitan dengan kemanan sistem operasi yang digunakan dalam jaringan
4. Sistem database, sistem dapat mengatur hak akses user

Pemberian Wewenang dan View

KONSEP VIEW adalah cara yang diberikan pada seorang pemakai untuk mendapatkan model database yang sesuai dengan kebutuhan perorangan

Database relational membuat pengamanan pada level :

- Relasi, seorang pemakai diperbolehkan atau tidak mengakses langsung suatu relasi
- View, seorang pemakai diperbolehkan atau tidak mengakses data yang terdapat pada view
- Read Authorization, data dapat dibaca tapi tidak boleh dimodifikasi
- Insert Authorozation, pemakai boleh menambah data baru, tetapi tidak dapat memodifikasi data yang sudah ada
- Update Authorization, pemakai boleh memodifikasi tetapi tidak dapat menghapus data
- Delete Authorization, pemakai boleh menghapus data
- Index Authorization, pemakai boleh membuat atau menghapus index
- Resource Authorization, mengizinkan pembuatan relasi – relasi baru
- Alternation Authorization, mengizinkan penambahan atau penghapusan atribute dalam satu relasi
- Drop Authorization, pemakai boleh menghapus relasi yang ada

11.6 INTEGRITY

Berarti memeriksa keakuratan dan validasi data

Beberapa Jenis Integrity :

1. **Integrity Konstains**, memberikan suatu sarana yang memungkinkan pengubahan database oleh pemakai berwenang sehingga tidak akan menyebabkan data inkonsistensi
2. **Integrity Rule** (pada basisdata relational), terbagi menjadi:
   - *Integrity Entity*, contoh: tidak ada satu komponen kunci primer yang bernilai kosong (null)
   - *Integrity Referensi*, suatu domain dapat dipakai sebagai kunci primer bila merupakan atribut tunggal pada domain yang bersangkutan

DAFTAR PUSTAKA

Connolly, Thomas and Begg, Carolyn. 2010. *Database Systems A Practical Approach to Design, Implementation, and Management Fifth Edition*. Boston: Pearson Education.

Hartono, Jogiyanto. 2005. Basis Data. Jakarta: Salemba Empat.

Indrajani. 2015. *Database Design (Case Study All in One).* Jakarta: PT Elex Media Komputindo.

Mulyanto, Agus.2009. Sistem Informasi Konsep dan Aplikasi. Yogyakarta: Andi.

Prasojo, Lantip Adi. 2014. Perancangan Database Sistem Informasi Manajemen Pendidikan Dengan Dbms Microsoft (Acces Dan Sql Server). Yogyakarta: UNY Press.