UNIVERSITAS
NUSA MANDIRI
Gedung Rektorat : Nusa Mandiri Tower Jl. Jatiwaringin Raya No. 2, Jakarta Timur 13620
Telp. (021) 28534471, 28534390 e-mail : rektorat@nusamandiri.ac.id

SURAT TUGAS

Nomor: 008/3.01/UNM/FTI/III/2024

Yang bertandatangan di bawah ini, Dekan Fakultas Teknologi Informasi Universitas Nusa Mandiri, menugaskan kepada:

No.	NIP	NAMA	MATA KULIAH
1	202104300	Elly Firasari, M.Kom	Sistem Basis Data

dalam Pembuatan Modul Mata Kuliah Program Studi Sistem Informasi Fakultas Teknologi Informasi Universitas Nusa Mandiri, dengan masa penugasan:

Tanggal : Maret s/d April 2024

Demikian hal ini disampaikan, sebagaimana semestinya. Atas perhatian dan kerjasamanya, kami ucapkan terima kasih.

Jakarta, 7 Maret 2024
Dekan Fakultas Teknologi Informasi

Anton, M.Kom

Tembusan:
1. Divisi SDM
2. Rektor
3. Wakil Rektor I Bidang Akademik
4. Wakil Rektor II Bidang Non Akademik
5. Kaprodi
6. Ybs.

Basis data merupakan elemen penting dalam pengelolaan informasi modern, yang memungkinkan organisasi untuk menyimpan, mengakses, dan mengelola data secara efisien mengelola basis data, serta teknik pengoptimalan kueri dan manajemen transaksi untuk memastikan integritas dan kinerja sistem.

SISTEM BASIS DATA

UNIVERSITAS NUSA MANDIRI

ELLY FIRASARI, M.KOM

KATA PENGANTAR

Puji syukur kami panjatkan kepada Tuhan Yang Maha Esa atas rahmat dan karunia-Nya, sehingga modul *Sistem Basis Data* ini dapat diselesaikan dengan baik. Modul ini disusun dengan tujuan untuk memberikan pemahaman yang komprehensif kepada pembaca mengenai konsep dan implementasi sistem basis data, serta memberikan keterampilan praktis yang diperlukan untuk mengelola basis data secara efisien.

Kami sangat berterima kasih kepada semua pihak yang telah berkontribusi dalam penyusunan modul ini. Ucapan terima kasih yang mendalam kami sampaikan kepada para pengajar, rekan-rekan sejawat, dan semua pihak yang telah memberikan masukan berharga dalam menyempurnakan isi dan materi yang terkandung di dalam modul ini. Tanpa dukungan dan bantuan dari berbagai pihak, penyusunan modul ini tidak akan bisa berjalan lancar.

Modul ini diharapkan dapat memberikan manfaat yang maksimal bagi pembaca, khususnya para mahasiswa, praktisi, dan siapa saja yang ingin memperdalam pengetahuan mengenai sistem basis data. Kami menyadari bahwa modul ini masih jauh dari sempurna, oleh karena itu kami sangat terbuka terhadap saran dan kritik yang konstruktif demi perbaikan di masa mendatang.

Akhir kata, kami mengucapkan terima kasih atas kepercayaan Anda dalam mempelajari modul ini. Semoga ilmu yang disajikan dalam modul ini dapat bermanfaat dan diterapkan dengan baik dalam dunia akademik maupun profesional.

Selamat belajar dan semoga sukses!

[Penulis / Tim Penyusun]

DAFTAR ISI

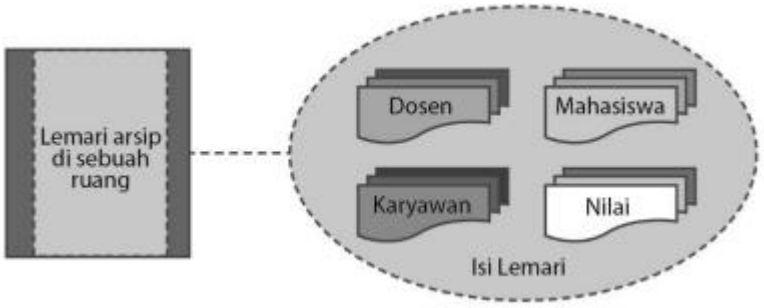
Halaman

BAB 1
KONSEP DASAR BASIS DATA (DATABASE)

Pada tahun 60-an, perkembangan tempat penyimpanan data masih dilakukan secara manual dan belum dikenal apa yang dinamakan basis data. Seluruh tempat penyimpanan disimpan dalam bentuk fisik sehingga dokumen yang tersimpan dirasakan belum optimal baik dari sisi tempat penyimpanan maupun dalam hal pencarian data. Seiring dengan perkembangan teknologi banyak perangkat lunak yang dikembangkan untuk menyimpan atau mengolah data secara elektronik seperti Microsoft Excel atau Apache Open Office Calc. Namun jika datanya berukuran besar maka diperlukan sebuah tempat penyimpanan data yang terintegrasi supaya penyimpanan dan pengolahan data menjadi lebih maksimal. Selanjutnya data yang tersimpan tersebut dapat diolah untuk menghasilkan informasi secara tepat, akurat, dan bermanfaat. Selain itu, dapat mempercepat upaya pelayanan kepada pelanggan dan membantu pengambilan keputusan atas suatu masalah berdasarkan informasi yang ada yang berasal dari basis data. Kehadiran basis data juga dapat meningkatkan kinerja dan daya saing sebuah organisasi.

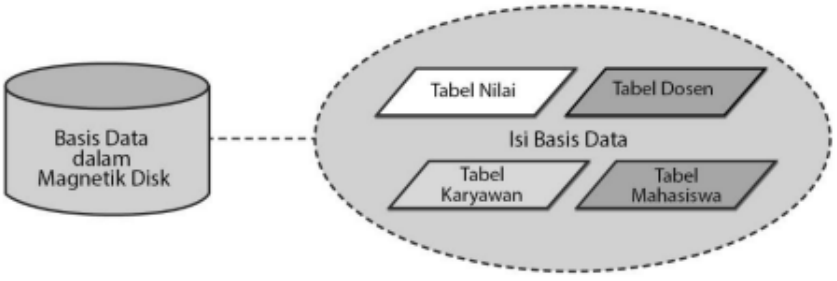
A. LATAR BELAKANG

Diperlukannya basis data dalam suatu perusahaan, pada dasarnya adalah untuk kemudahan dan kecepatan dalam pengambilan data. Untuk lebih jelasnya dapat dilihat pada Gambar 1.1 dan Gambar 1.2.



Sumber : Fathansyah (2015), diolah penulis

Gambar 1.1
Lemari Berkas



Gambar 1.2
Basis Data Magnetik

Dari gambar tersebut, terlihat perbedaan antara basis data dan lemari arsip di mana setiap rak dalam lemari tersebut dapat menyimpan dokumen-dokumen manual yang terdiri dari lembaran-lembaran kertas. Masalah yang dihadapi pada lemari arsip adalah kelambatan dalam proses penelusuran data pada lemari arsip tersebut. Misalkan ingin mencari arsip untuk pegawai tertentu. Untuk menemukan hasilnya, akan membutuhkan waktu yang lama. Hal ini disebabkan proses pencarian harus mencari lembaran–lembaran yang ada pada dokumen tersebut dan dapat menyebabkan waktu pencarian yang kurang efisien. Berikut adalah beberapa alasan mengapa sebuah organisasi atau perusahaan memerlukan sebuah basis data, diantaranya:

1. **Membantu Pengelolaan Data yang Besar**

   Basis data dapat menyimpan dan membantu mengelola data dalam jumlah yang besar secara rinci dan terintegrasi. Hal ini sangat tidak mungkin jika menggunakan Perangkat lunak spreadsheet seperti Microsoft Excel atau Apache OpenOffice Calc karena kapasitas penyimpanan dan kecepatan mengolah data dalam Microsoft Excel dan Apache OpenOffice sangat terbatas.

2. **Akurasi Data**

   Sebuah basis data secara umum dapat menjamin akurasi data. Hal tersebut dikarenakan basis data memiliki fitur constraints dan default value check. Constrains adalah aturan-aturan di dalam tabel basis data yang dapat mencegah penghapusan atau perubahan data dari suatu tabel karena data dalam tabel tersebut mempunyai keterkaitan dengan data pada tabel lain. Sedangkan default value check adalah proses cek jika data yang dimasukkan ke dalam basis data tidak mempunyai nilai (null) maka nilai default yang akan digunakan

3. **Mudah dalam Proses Manipulasi Data**

   Sebuah perangkat lunak basis data, memiliki fitur yang dapat memudahkan dalam melakukan proses manipulasi data (proses insert/update/delete). Salah satunya dengan menggunakan Data Manipulation Languages (DML) yang termasuk ke dalam Structured Query Language (SQL).

4. **Keamanan Data**

   Perangkat lunak basis data memiliki fitur keamanan. Hal tersebut bertujuan untuk menjaga data dari hal-hal yang tidak diinginkan. Sebagai contoh misalnya sebelum pengguna mendapatkan data yang diinginkan, pengguna harus melakukan login terlebih dahulu, sehingga data lebih aman.

5. **Integritas Data**

   Sebuah basis data dapat menjamin integritas data. Karena basis data memiliki fitur constrains, maka integritas data dalam basis data dapat terjamin. Sebagai contoh jika terjadi perubahan data di dalam Tabel A, maka tabel yang memiliki keterkaitan dengan Tabel A akan mengikuti perubahan tersebut atau basis data akan mencegah perubahan pada Tabel A jika diatur demikian. Demikian juga untuk proses penambahan data maupun penghapusan data. Dengan adanya hal tersebut integritas data dapat terjamin.

B. PENGERTIAN BASIS DATA

Berbicara tentang basis data dapat diartikan bahwa seluruh data yang disimpan dalam sebuah basis data ditempatkan pada masing-masing table/file sesuai dengan fungsinya. Dengan tersimpannya data dalam basis data tersebut, maka akan dengan mudah dapat melakukan penelusuran data yang diinginkan sehingga berdampak pada waktu pencarian yang lebih efisien. Di dalam suatu media penyimpanan (misalnya harddisk), dapat ditempatkan lebih dari 1(satu) basis data secara elektronik. Namun, tidak semua bentuk penyimpanan data yang disimpan secara elektronik dikatakan basis data karena ketika menyimpan dokumen di dalam sebuah harddisk, harddisk tersebut dapat berisi data file teks dari program pengolahan kata, spreadsheet, dan lainnya. Yang ditekankan dalam basis data adalah pengaturan, pemilahan, pengelompokan, dan pengorganisasian data yang disimpan sesuai dengan fungsinya. Hal tersebut bisa berbentuk sejumlah file, table terpisah, atau dalam bentuk pendefinisian kolom (field) data dalam setiap file atau table tersebut.

- **Pengertian Basis Data**

Basis Data berasal dari kata Basis dan Data. Adapun pengertian dari kedua kata tersebut adalah sebagai beriku

1. **Basis** dapat diartikan sebagai markas atau gudang atau tempat bersarang atau tempat berkumpul.
2. **Data** dapat diartikan sebagai representasi fakta dunia nyata yang mewakili suatu objek seperti manusia (pegawai, siswa, pembeli, dan pelanggan), barang, hewan, peristiwa, konsep, keadaan, dan sebagainya. Objek tersebut direkam dalam bentuk angka, huruf, simbol, teks, gambar, bunyi, atau kombinasinya.

Basis Data sebagai satu kesatuan dapat didefinisikan sebagai berikut.

1. Himpunan kelompok data yang saling berhubungan dan terorganisasi dengan baik agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah.
2. Kumpulan data yang saling berhubungan dan disimpan dengan baik secara bersama-sama tanpa pengulangan (redudansi) yang tidak diperlukan.
3. Kumpulan file, tabel, atau arsip yang saling berhubungan dan disimpan dalam satu media penyimpanan elektronik. Kumpulan file ini selanjutnya disebut Tabel (Table) sebagai komponen utama untuk membangun basis data. (Fathahansyah, 2015)

Dalam beberapa literatur, basis data telah didefinisikan dengan cara yang berbeda. Salah satu definisi yang cukup lengkap dan baik tentang istilah basis data adalah yang diberikan oleh James Martin (1975) dalam buku Sistem Basis Data (Edhy Sutanta, 2004, h. 17) sebagai berikut:

> **“A database may be defined as a collection of interrelated data stored together without harmful or unnecessary redundancy to serve one or more application in an optimal fashion; the data are stored so that they are independent of programs its used the data; a common and controlled approach its used in adding new data and in modifying and retrieving exiting data whithin the database”.**

Basis Data dapat dipahami sebagai suatu kumpulan data terhubung yang disimpan secara bersama-sama pada suatu media. Walaupun disimpan secara bersamasama dan saling terhubung, kumpulan data tersebut tersimpan tanpa saling tumpang tindih satu sama lain atau tidak terjadi kerangkapan data. Namun, jika pun terjadi kerangkapan data maka kerangkapan data tersebut harus terjadi seminimal mungkin dan dapat terkontrol. Beberapa kondisi data di dalam suatu basis data diantaranya:

1. data disimpan dengan cara-cara tertentu sehingga memudahkan ketika akan digunakan atau ditampilkan kembali.
2. data dapat digunakan oleh satu atau beberapa program aplikasi secara optimal.
3. data disimpan tanpa mengalami ketergantungan dengan program-program yang akan menggunakannya.
4. data disimpan sedemikian rupa sehingga proses penambahan, pengambilan, dan modifikasi data dapat dilakukan dengan mudah dan terkontrol.

Dari definisi tersebut dapat disimpulkan bahwa pengertian basis data adalah koleksi terpadu dari data yang saling berkaitan dan dirancang untuk memenuhi kebutuhan informasi suatu organisasi. Masing–masing table/file di dalam basis data tersebut berfungsi untuk menampung atau menyimpan data dimana data-data tersebut saling berhubungan dengan satu dengan yang lain. Dari pengertian tersebut dapat dikatakan bahwa basis data memiliki beberapa kriteria penting antara lain:

1. beorientasi pada data.
2. data dapat digunakan oleh pemakai yang berbeda-beda atau beberapa program aplikasi tanpa perlu mengubah basis data.
3. data dalam basis data dapat berkembang dengan mudah baik volume maupun strukturnya.
4. data yang ada dapat memenuhi kebutuhan sistem-sistem baru secara mudah.
5. data dapat digunakan dengan cara yang berbeda-beda.
6. kerangkapan data minimal.

C. MANFAAT DAN TUJUAN BASIS DATA

Manfaat basis data adalah untuk pengelolaan data dalam memudahkan atau menemukan kembali data yang dicari dengan cepat. Tujuan Basis Data antara lain sebagai berikut:

1. **Kecepatan dan kemudahan (Speed).** Dengan basis data dapat menyimpan data atau melakukan perubahan, penghapusan, penambahan, dan pemanggilan kembali data yang tersimpam dengan cepat dan mudah.
2. **Efisiensi ruang penyimpanan (Space).** Dengan basis data penggunaan ruang penyimpanan data dapat dilakukan dengan melakukan meminimalisasi jumlah pengulangan data dan dengan menerapkan sejumlah pengkodean.
3. **Keakuratan (Accuracy).** Dengan memanfaatkan pengkodean atau pembentukan relasi antar data, penerapan aturan atau batasan tipe data dapat diterapkan dalam basis data yang berguna untuk menentukan keakuratan saat input data atau penyimpanan data

4. **Keamanan (Security).** Sejumlah sistem (aplikasi) pengelolaan basis data tidak menerapkan aspek keamanan dalam penggunaannya. Akan tetapi, untuk sistem yang besar dan serius, aspek keamanan menjadi hal yang penting untuk diterapkan. Dengan begitu, sistem dapat menentukan siapa yang boleh menggunakan basis data dan menentukan jenis operasi-operasi apa saja yang boleh dilakukan.
5. **Terpeliharanya keselarasan data (Consistent).** Apabila ada perubahan data pada aplikasi yang berbeda, secara otomatis perubahan itu berlaku untuk keseluruhan.
6. **Kebersamaan pemakaian** (**Sharebility**). Data dapat dipakai secara bersama-sama oleh beberapa program aplikasi saat bersamaan.
7. Dapat diterapkan standarisasi (Standardization). Dengan adanya pengontrolan yang terpusat, basis data dapat menerapkan standarisasi data yang disimpan sehingga memudahkan pemakaian, distribusi, maupun pertukaran data.
8. **Ketersediaan (Availability**). Basis data dapat memilah data utama atau master, transaksi, data history hingga data kedaluwarsa. Data yang jarang atau tidak digunakan lagi dapat diatur untuk dipisahkan dari sistem basis data yang aktif.
9. **Kelengkapan (Completeness).** Kelengkapan sebuah data bersifat relatif, dalam sebuah basis data penilaian kelengkapan data sangat bergantung pada pengguna sehingga penilaian tidak selalu sama

D. KELEBIHAN DAN KEKURANGAN BASIS DATA

Kelebihan:

1. Dapat meningkatkan kemandirian data. Sebuah basis data dapat digunakan untuk bermacam-macam program aplikasi tanpa harus mengubah format data yang sudah ada.
2. Konsistensi data. Konsistensi data di dalam basis data dilakukan dengan cara data disimpan hanya sekali dalam basis data sehingga jika terjadi perubahan pada nilai data tersebut, perubahan hanya dilakukan satu kali dan nilai baru tersebut akan tersedia untuk semua pengguna.
3. Meningkatkan aksesibilitas terhadap data dan respon yang lebih baik. Dengan basis data maka aksesibilitas data dan respon akan lebih baik. Hal tersebut dapat dicapai dengan integrasi data yang melewati batasan-batasan departemen dalam organisasi sehingga data dapat langsung diakses oleh pengguna.
4. Pengendalian terhadap kerangkapan data. Data dalam sebuah basis data dilakukan penyimpanan dengan cara disimpan satu kali. Hal ini mengurangi kerangkapan data dan mengurangi biaya untuk tempat penyimpanan.
5. Meningkatkan keamanan data. Keamanan basis data dapat melindungi basis data dari pengguna yang tidak memiliki otorisasi. Basis data dapat menentukan batasan-batasan pengaksesan data, misalnya dengan memberikan password dan pemberian hak akses bagi pemakai (misalnya untuk hak akses dalam proses update, delete, insert, maupun select).
6. Memperbaiki integritas data. Intergritas data mengacu pada validitas dan konsistensi dari data yang disimpan. Integritas biasanya diekspresikan dalam batasan (constraints)

yang merupakan aturan yang konsisten dan tidak dapat dilanggar. Jika kerangkapan data dapat dikontrol dan kekonsistenan data dapat dijaga, maka data menjadi akurat.

7. Data dapat dipakai secara bersama-sama. Data yang ada pada basis data menjadi milik seluruh organisasi dan dapat dipakai secara bersama oleh pengguna yang berwenang pada saat bersamaan.
8. Memperoleh lebih banyak informasi dari data yang sama. Pengguna basis data dapat memperoleh informasi selain dari informasi rutin yang dikelolanya karena semua data lain berada dalam basis data yang sama. Dengan demikian, kebutuhan akan informasi selain dari informasi rutin dapat terpenuhi.

Kekurangan:

1. Biayanya dapat menjadi sangat mahal karena menyangkut biaya-biaya untuk pembelian sekaligus perawatan hardware dan software. Selain itu, terdapat juga biaya tambahan untuk untuk penyimpanan (storage), jaringan (network), dan lain-lain.
2. Rumit. Perancang, pengembang, Data Base Administator (DBA), dan pengguna akhir harus memahami secara rinci dan mendalam tentang fungsi basis data yang ditangani agar dapat mengambil manfaat dari basis data. Kegagalan dalam memanfaatkannya dapat menyebabkan kerugian yang cukup besar bagi organisasi atau perusahaan.
3. Tambahan biaya konversi. Diperlukan biaya yang besar untuk berpindah dari aplikasi atau sistem yang lama ke dalam sistem basis data yang baru. Selain itu, diperlukan pula biaya untuk pelatihan staf dalam menggunakan sistem yang baru ini serta tambahan biaya untuk mempekerjakan staf khusus seperti DBA, dan lain-lain.

E. KOMPONEN SISTEM BASIS DATA

Terdapat 4 komponen pokok dari sistem basis data:

1. DATA, dengan ciri-ciri :
   a. 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)
   b. 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:

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

2. 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:
   a. Komputer (satu untuk sistem stand-alone atau lebih dari satu untuk sistem jaringan)
   b. Memori sekunder on-line (Harddisk)

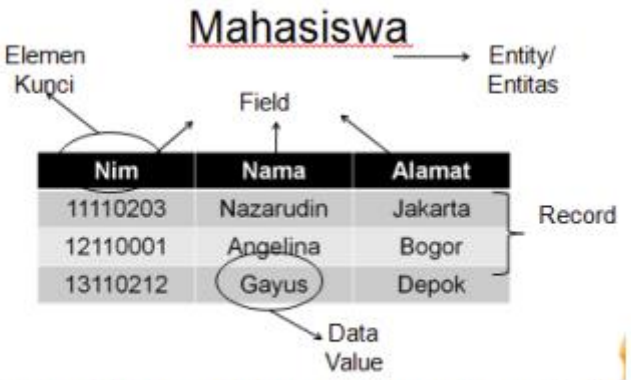
c. Memori sekunder off-line (Tape atau Removeble Disk) untuk backup data
d. Media/perangkat komunikasi (untuk sistem jaringan)

3. Perangkat Lunak (SOFTWARE) Berfungsi sebagai perantara (interface) antara pemakai dengan data phisik pada database, dapat berupa :
   a. Database Management System (DBMS)
   b. Program-program aplikasi & prosedur-prosedur

4. 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 :
   a. Database Administrator (DBA), yang membuat basis data dan mengontrol akses ke basis data.
   b. Programmer, yang membuat aplikasi basis data yang digunakan oleh DBA dan pemakai akhir.
   c. Pemakai akhir (End user) yang melakukan penambahan, penghapusan, pengubahan, dan pengaksesan data

Istilah-istilah Dalam Sistem Basis Data a.

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 buki 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

A. PENGERTIAN SISTEM BASIS DATA DAN KOMPONEN SISTEM

Berbeda dengan Basis Data (database), sistem basis data dapat diartikan sebagai suatu sistem yang di dalamnya terdiri dari koleksi data atau suatu kumpulan data yang saling berhubungan dan memungkinkan berbagai program untuk mengakses dan memanipulasi data tersebut. Sistem basis data juga merupakan suatu sistem yang menyusun dan mengelola data suatu organisasi, sehingga mampu menyediakan informasi yang diperlukan oleh pemakai.

Terdapat beberapa komponen dalam sebuah sistem basis data diantaranya:

1. Perangkat keras (hardware).
2. Sistem operasi (operating system).
3. Basis data (database).
4. DBMS (database management system), merupakan perangkat lunak (software) yang digunakan untuk menentukan bagaimana data tersebut dapat terorganisasi, tersimpan, diubah serta diambil kembali. DBMS ini pun yang menerapkan suatu mekanisme sebagai pengamanan data secara bersamaan, konsistensi data, dan semacamnya.
5. Pengguna (user). Pengguna ini dapat dikategorikan menjadi pengguna akhir atau end user, pemrogram aplikasi dan administrator basis data atau DBA (Database Administrator).
6. Program aplikasi (application program) adalah perangkat lunak yang ditulis atau dikembangkan oleh Programmer atau pemrogram aplikasi dan kemudian digunakan oleh end user atau pengguna akhir.

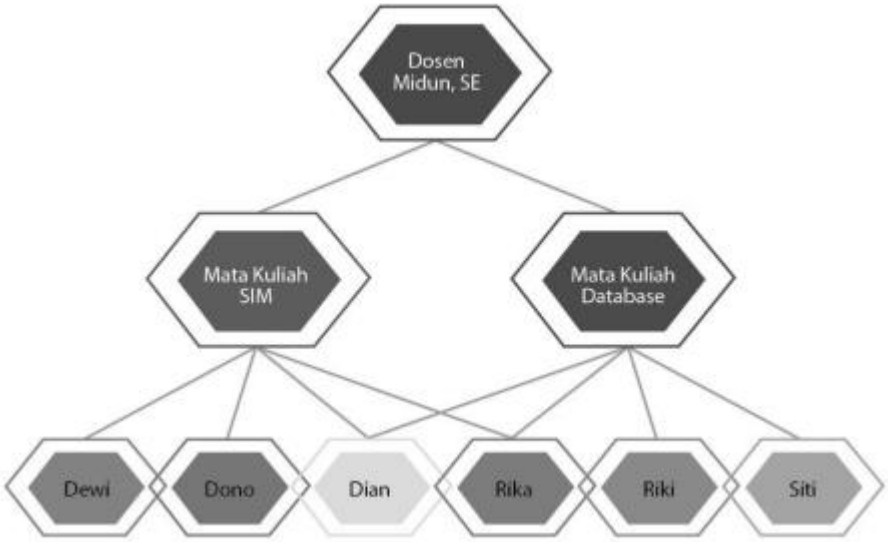
B. KONSEP DAN SEJARAH DATABASE MANAGEMENT SYSTEM (DBMS)

Database Management System (DBMS) merupakan paket program (Software) yang digunakan untuk membuat dan mengelola basis data. DBMS dibuat agar memudahkan dan mengefisienkan proses input, update, delete, restore, view, dan pengambilan informasi terhadap basis data. Tujuan utama DBMS adalah menyediakan lingkungan yang mudah dan nyaman bagi pengguna dalam mengambil, menyimpan data dan informasi. Software yang tergolong ke dalam DBMS antara lain, Microsoft SQL, MySQL, Oracle, MS. Access, dan lain-lain.

Sejarah Database Management System (DBMS):

Tahun 1960 Network Database Sejarah sebuah basis data diawali tahun 1960. Pada awal tahun tersebut, Charles Bachman membuat sebuah desain DBMS pertama yang disebut dengan penyimpanan data terintegrasi. Desain DBMS tersebut berbentuk sebuah model jaringan basis data (network database). Model jaringan basis data tersebut distandarisasi oleh

Conference on Data System Language (CODASYL). Gambar 1.4 merupakan contoh dari network database.



Gambar 1.4
Network Database

Network database terdiri dari kumpulan record yang dihubungkan melalui pointer yang membentuk relasi antar record. Model ini tidak memungkinkan banyak relasi namun mudah dalam menyisipkan data.

C. 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:

1. Mengurangi pengulangan data
2. Mencapai independensi data
3. Mengintegrasikan data beberapa file
4. Mengambil data dan informasi dengan cepat
5. Meningkatkan keamanan

Kerugian DBMS:

1. Perangkat lunak yang mahal
2. Konfigurasi perangkat keras yang besar
3. Mempekerjakan dan mempertahankan staf DBA

Abstraksi Data

DBMS memiliki tujuan untuk menyediakan interface kepada pengguna. Abstraksi data merupakan tingkatan-tingkatan pengguna dalam memandang bagaimana sebenarnya data diolah dalam sebuah sistem basis data sehingga menyerupai kondisi yang sebenarnya dihadapi oleh pengguna dalam kehidupan sehari-hari. Sebuah DBMS seringkali menyembunyikan secara rinci tentang bagaimana sebuah data disimpan dan dipelihara (diolah) dalam sebuah sistem basis data. Hal ini bertujuan untuk memudahkan pengguna

dalam menggunakan DBMS tersebut. Oleh karena itu, pengguna seringkali melihat perbedaan data sebelumnya dengan data yang tersimpan secara fisik.

Terdapat 3 level abstraksi yaitu:

1. Level Fisik (Physical Level)
   Level fisik merupakan lapisan terendah. Lapisan ini menjelaskan bagaimana (how) data sesungguhnya disimpan. Pada lapisan inilah struktur data dijabarkan secara terperinci.
2. Level Logik / Konseptual (Conceptual Level)
   Level konseptual lebih tinggi dari lapisan fisik. Lapisan ini menjabarkan data apa (what) saja yang sesungguhnya disimpan pada basis data dan juga menjabarkan hubungan-hubungan antar data secara keseluruhan. Seorang pengguna dalam level ini dapat mengetahui bahwa data mahasiswa disimpan pada tabel mahasiswa, tabel KRS, tabel transkrip, dan lain sebagainya. Level ini biasa dipakai oleh DBA (Database Administrator).
3. Level Pandangan (View Level)
   Level pandangan merupakan lapisan tertinggi pada abstraksi data. Pada lapisan ini pengguna hanya mengenal struktur data sederhana yang berorientasi pada kebutuhan pengguna. Data yang dikenal oleh setiap pengguna bisa berbeda-beda dan barangkali hanya mencakup sebagian dari basis data. Misalnya, bagian keuangan hanya membutuhkan data keuangan, jadi yang digambarkan hanya pandangan terhadap data keuangan saja. Begitu juga dengan bagian akuntansi, hanya membutuhkan data akuntansi. Jadi, tidak semua pengguna basis data membutuhkan seluruh informasi yang terdapat dalam basis data tersebut.

D. 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 danVjalur pengaksesan yang didukung sistem manajemen database, Tipe interface dan programmer, Tipe bahasa query
   b. Faktor Ekonomi dan Politik organisai
      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

A. DEFINISI MODEL DATA

Menurut (Fathansyah, 2012) model data dapat didefinisikan sebagai kumpulan perangkat konseptual untuk menggambarkan data, hubungan data, semantic (makna) data dan batasan data. Pada umumnya sebuah model dinyatakan dalam bentuk diagram yang dibuat di awal akan lebih mudah untuk dievaluasi maupun dianalisis untuk kemudian dilakukan perbaikanperbaikan untuk mendapatkan sebuah model data yang lebih permanen dan lebih mendekati kenyataan sesungguhnya.

Model data merupakan suatu cara untuk menjelaskan tentang data-data yang tersimpan dalam basis data dan bagaimana hubungan antar data tersebut untuk para pemakai secara logic. Model data juga dapat diartikan sebagai kumpulan perangkat konseptual untuk menggambarkan data, hubungan data, semantic (makna) data dan batasan data

B. JENIS-JENIS MODEL DATA

Model data terbagi menjadi dua yaitu:

1. Model Data Berbasis Objek

Model data berbasis objek menggunakan konsep entitas, atribut dan hubungan antar entitas (relationship). Entity adalah sesuatu apa saja yang ada didalam sistem, nyata maupun abstrak dimana data tersimpan atau dimana terdapat data. Entitas diberi nama dengan kata benda dan dapat dikelompokkan dalam empat jenis nama, yaitu nama orang, benda, lokasi, kejadian (terdapat unsur waktu didalamnya). Atribut adalah relasi fungsional dari satu objek set ke objek set yang lain. Sedangkan, Relationship adalah hubungan alamiah yang terjadi antara entitas.

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**.

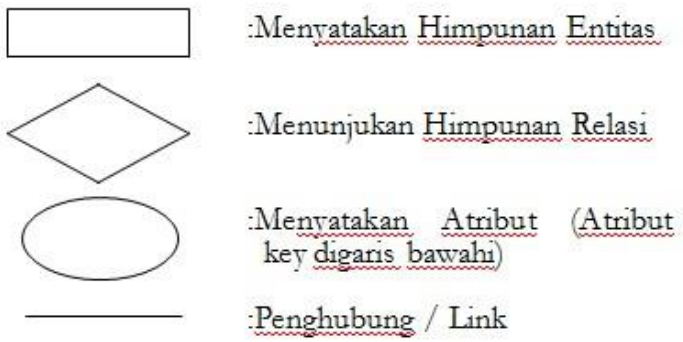
Diagram Entity-Relationship (Diagram E-R)

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
:

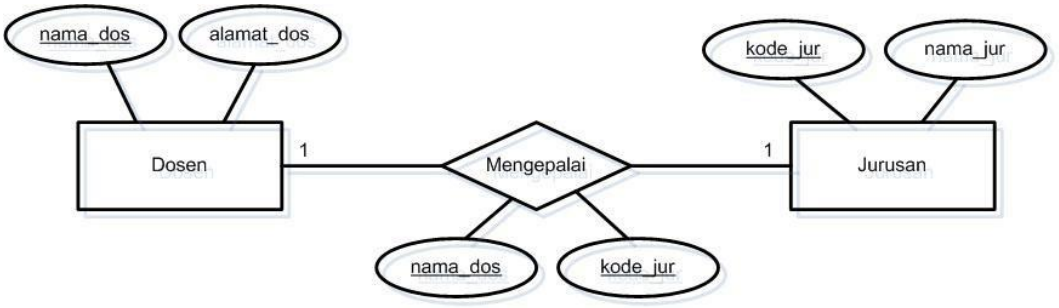


Dalam Diagram E-R aturan terpenting adalah Kardinalitas relasi/**Mapping Cardinalities** yang menentukan jumlah entity yang dapat dikaitkan dengan entity lainnya melalui relationship-set.

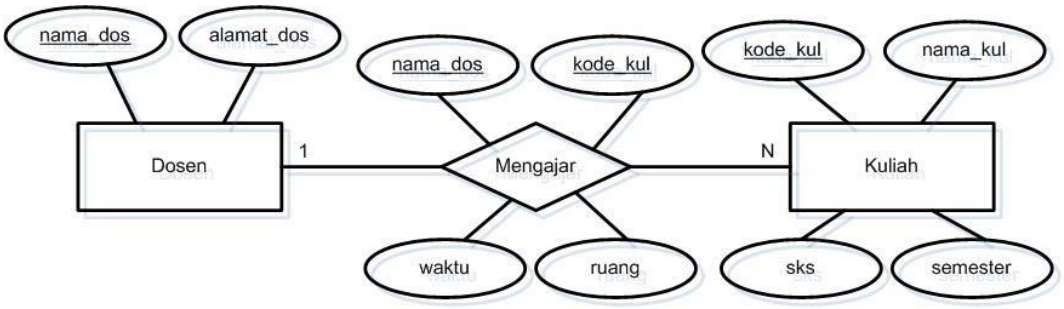
Jenis Mapping Cardinalities:

- Relasi satu ke satu (*one-to-one*)

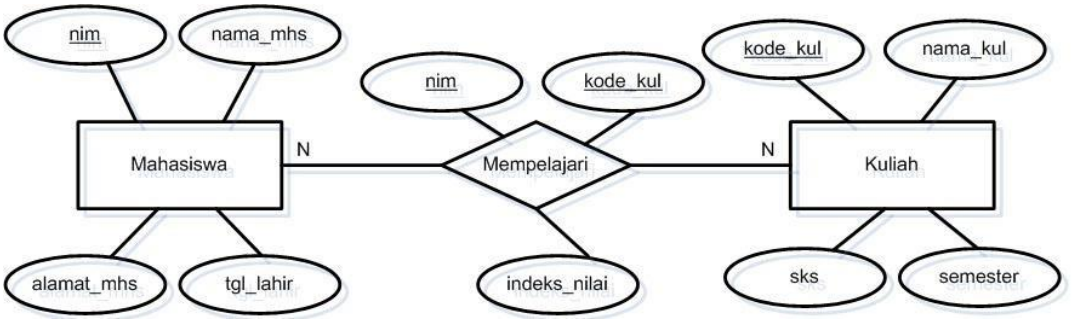
- Relasi satu ke banyak (*one-to-Many*)
- Relasi banyak ke banyak (*many-to-many*)



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

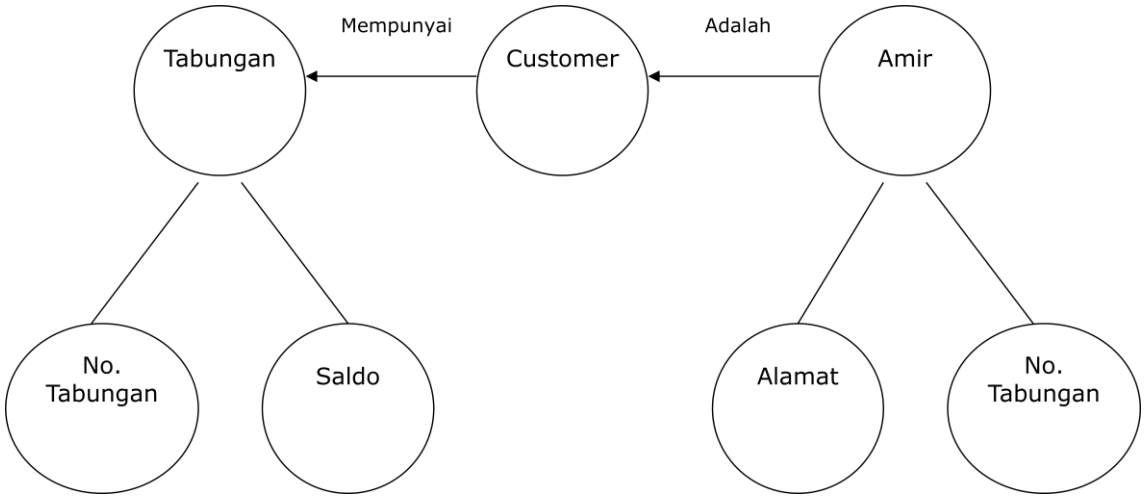
Hampir sama dengan Entity Relationship model dimana relasi antara objek dasar tidak dinyatakan dengan simbol tetapi menggunakan kata-kata (Semantic). Sebagai

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

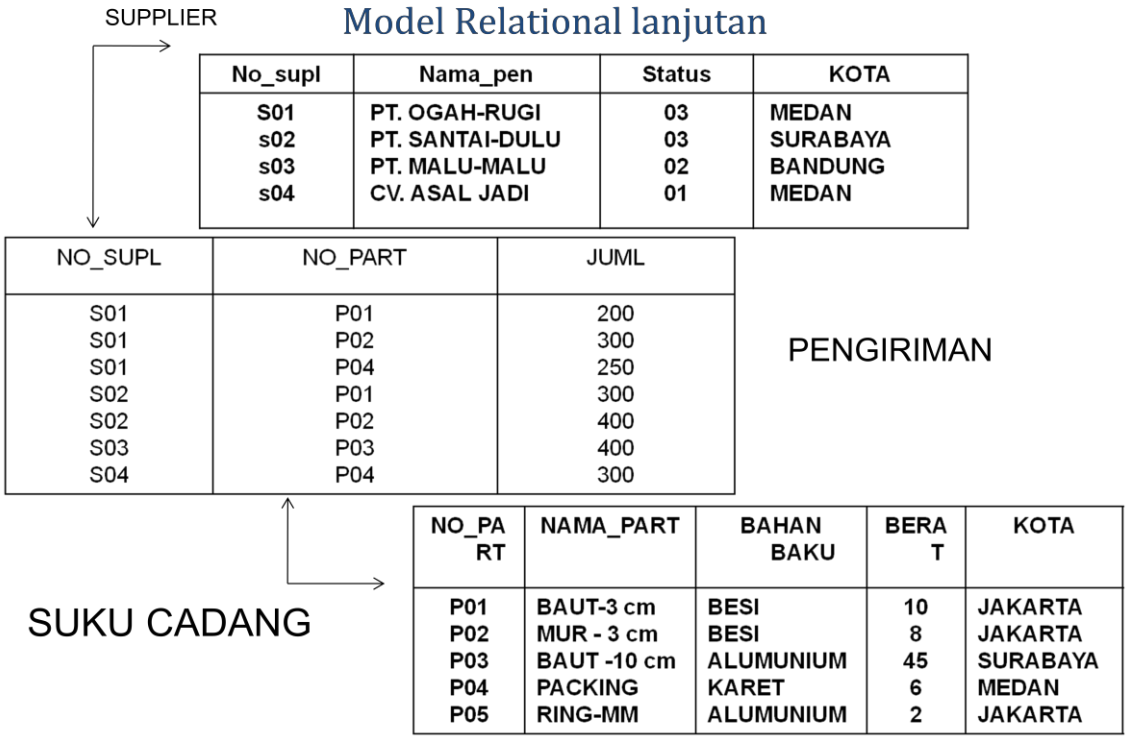
Terdapat 3 data model pada model data berbasis record:

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:

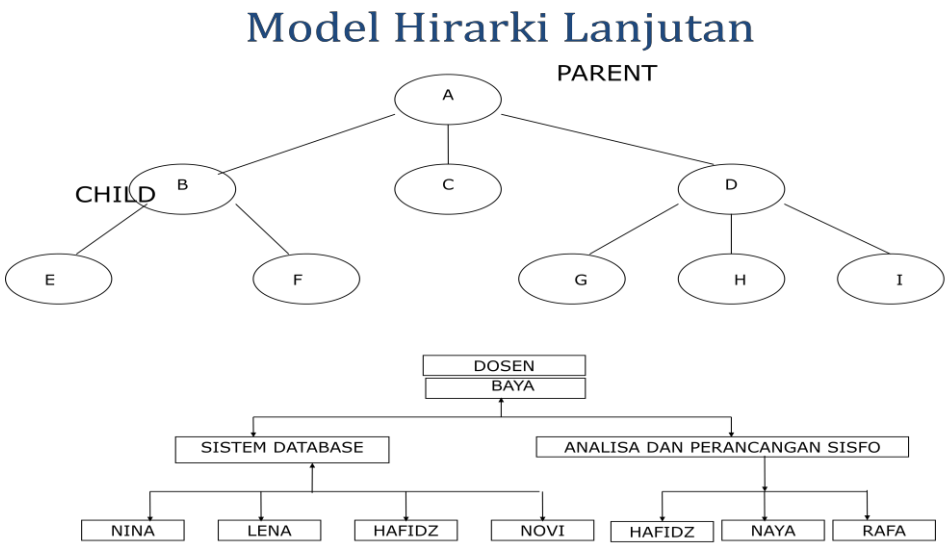
- Supllier
- Suku_cadang
- Pengiriman



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 .

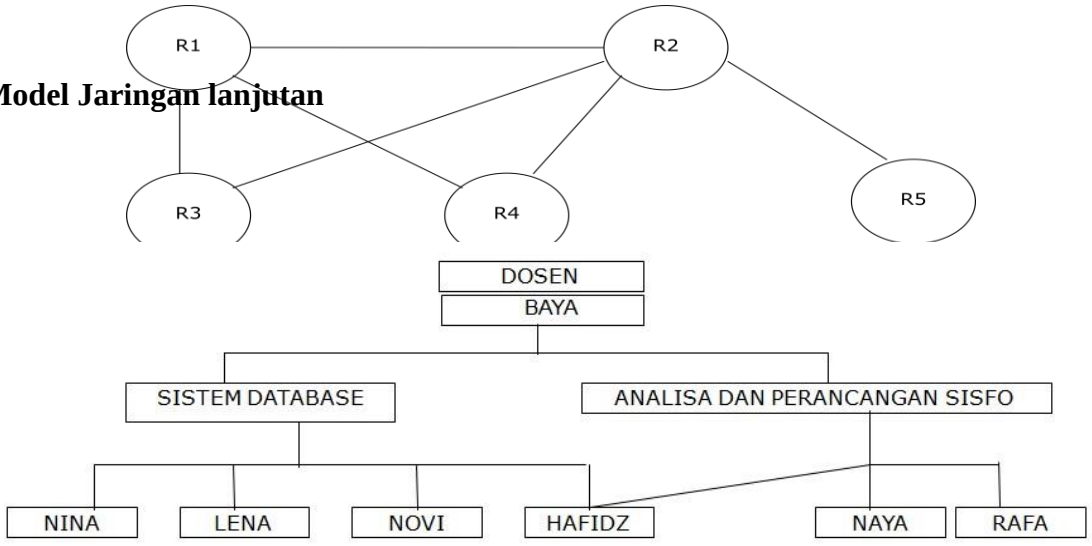


Model Jaringan

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)

A. Definisi Entity Relationship Diagram (ERD)

Menurut (Sukamto & Shalahuddin, 2018), Entity Relationship Diagram (ERD) adalah bentuk paling awal dalam melakukan perancangan basis data relasional. jika menggunakan OODBMS maka perancangan ERD tidak diperlukan. Menurut (Fathansyah, 2012), ERD adalah model entity relationship yang berisi komponenkomponen himpunan entitas dan himpunan relasi yang masing-masing dilengkapi dengan atribut-atribut yang merepresentasikan seluruh fakta 'dunia nyata' yang kita tinjau.

B. Komponen Entity Relationship Diagram (ERD)

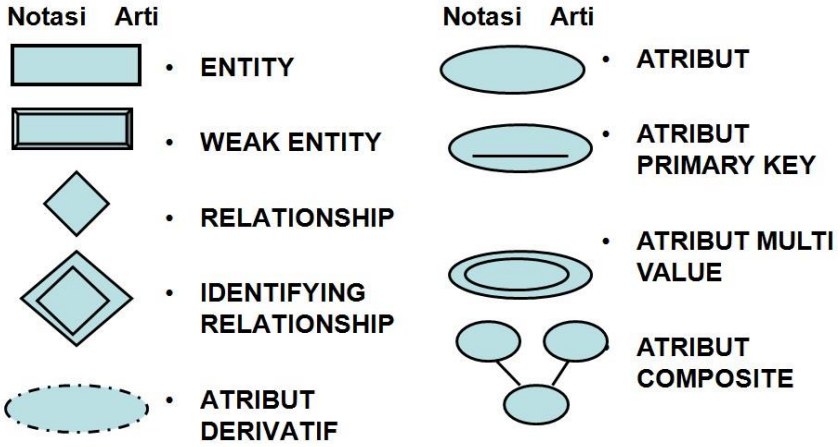
Ada beberapa komponen pada ERD diantaranya:

1. Entitas Entitas adalah suatu kumpulan object atau sesuatu yang dapat dibedakan atau dapat diidentifikasikan secara unik. Dan kumpulan entitas yang sejenis disebut dengan entity set. Entity Set terbagi menjadi dua, yaitu:

a. Strong entity set, yaitu entity set yang satu atau lebih atributnya digunakan oleh entity set lain sebagai key.
b. Weak Entity set, yaitu entity set yang bergantung terhadap strong entity set.

2. Atribut Atribut adalah kumpulan elemen data yang membentuk suatu entitas. Adapun jenis-jenis atribut:
   a. Atribut kunci, adalah atribut yang digunakan untuk menentukan suatu entity secara unik
   b. Atribut simple, adalah atribut yang bernilai tungga.
   c. Atribut multi value, adalah atribut yang memiliki sekelompok nilai untuk setiap instan entity.
   d. Atribut composit, adalah suatu atribut yang terdiri dari beberapa atribut yang lebih kecil yang mempunyai arti tertentu.
   e. Atribut derivatif, adalah suatu atribut yg dihasilkan dari atribut yang lain.
   f. Relationship Relationship merupakan hubungan yang terjadi antara satu entitas atau lebih.
   g. Participati on Constraint menjelaskan apakah keberadaan suatu entity tergantung pada hubungannya dengan entity lain. Terdapat dua macam participation constrain yaitu: a. Total participation constrain, adalah keberadaan suatu entity tergantung pada hubungannya dengan entity lain. Didalam diagram ER digambarkan dengan dua garis penghubung antar entity dan relationship. b. Partial participation, adalah keberadaan suatu entity tidak tergantung pada hubungan dengan entity lain. Didalam diagram ER digambarkan dengan satu garis penghubung. 4. Indicator Type Ada beberapa indicator type, diantaranya: a. Indicator Asosiatif Object, berfungsi sebagai suatu objek dan suatu relationship. b. Indicator Super Tipe Indicator tipe super tipe, terdiri dari suatu object dan satu subkategori atau lebih yang dihubungkan dengan satu relationship yang tidak bernama.

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.

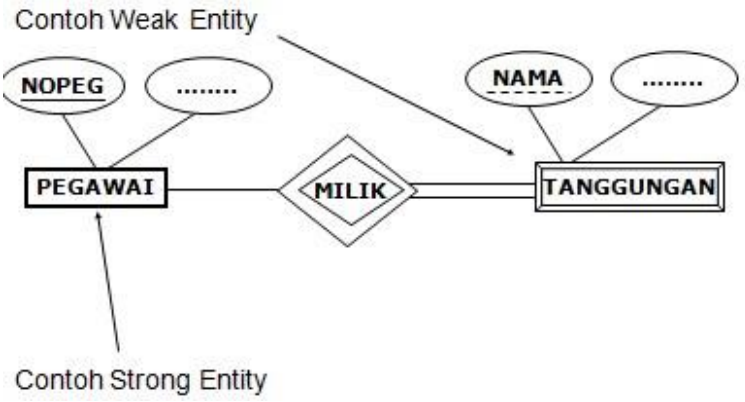
2. Relationship yaitu hubungan yang terjadi antara satu entitas atau lebih.
3. Atribut, kumpulan elemen data yang membentuk suatu entitas.
4. Indicator tipe terbagi 2 yaitu :
   a. Indicator tipe asosiatif object b. Indicator tipe super tipe

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}

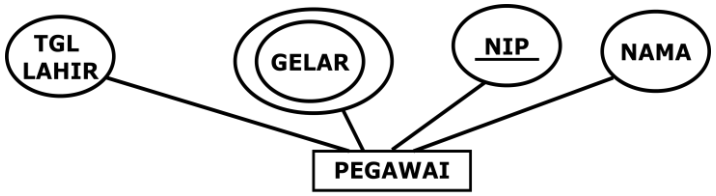


Jenis-Jenis Atribut

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

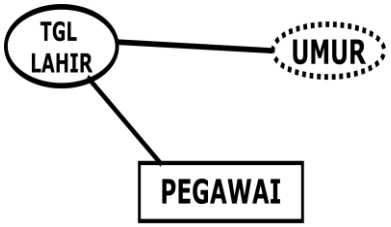
   Pada gambar dibawah ini, yang menjadi atribut key adalah NIP.

   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.

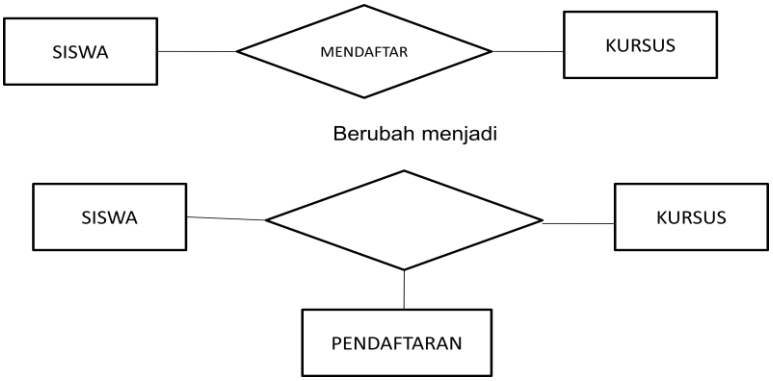
Contoh Participation Constraint

a. TOTAL PARTICIPATION

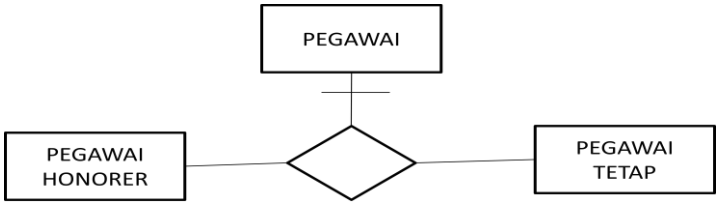
b. PARTIAL PARTICIPATION

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.

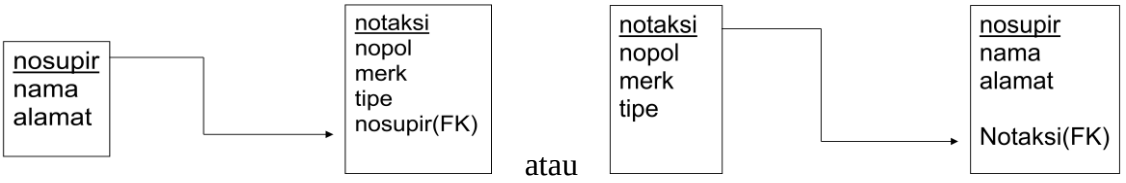
Menentukan Kardinalitas, Jumlah Tabel dan Foreign Key (FK)

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)

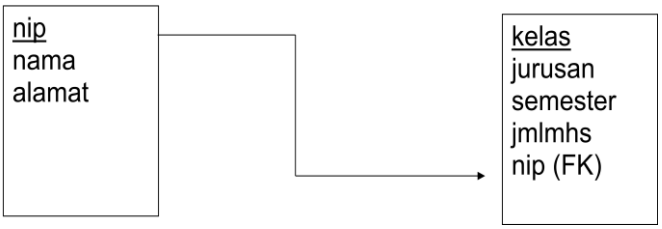
Gambar di atas menunujukan relasi dengan kardinalitas 1-M, karena:

1 Dosen bisa membimbing banyak Kelas, dan

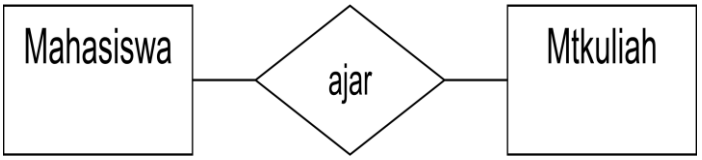
1 Kelas hanya dibimbing oleh 1 Dosen.
Relasi 1-M akan membentuk 2 tabel:

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:

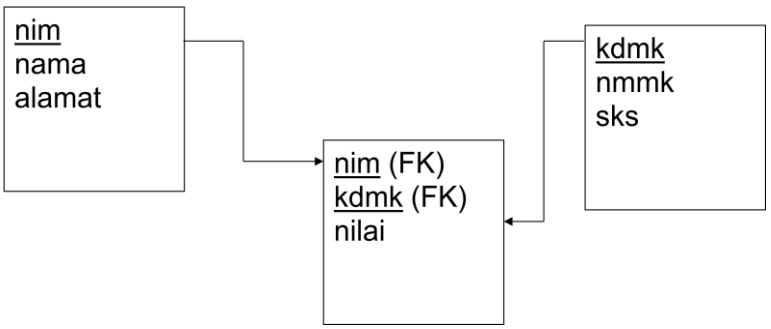
1 Mahasiswa bisa belajar banyak Mata Kuliah, dan

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:


nim
nama
alamat
kdmk
nmmk
sks
nim (FK)
kdmk (FK)
nilai

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

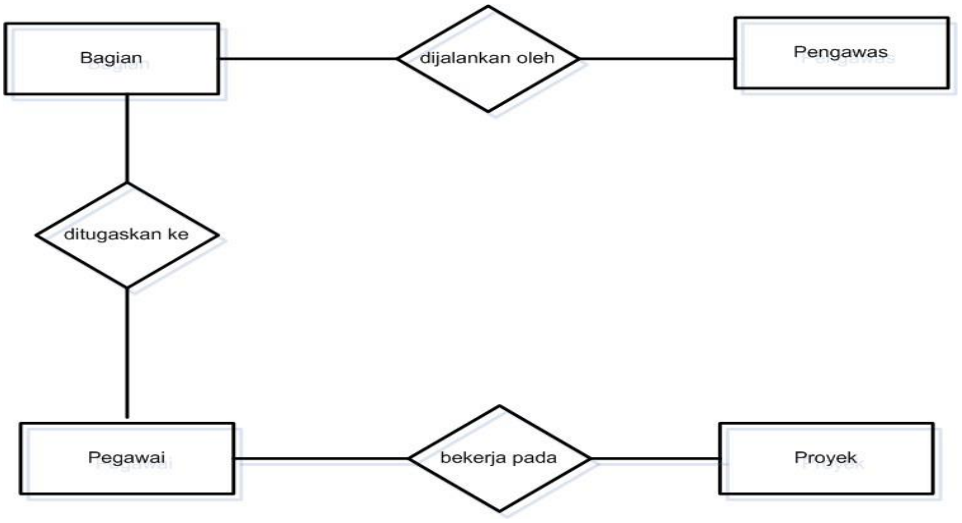
Langkah 1 : Menentukan Entitas

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

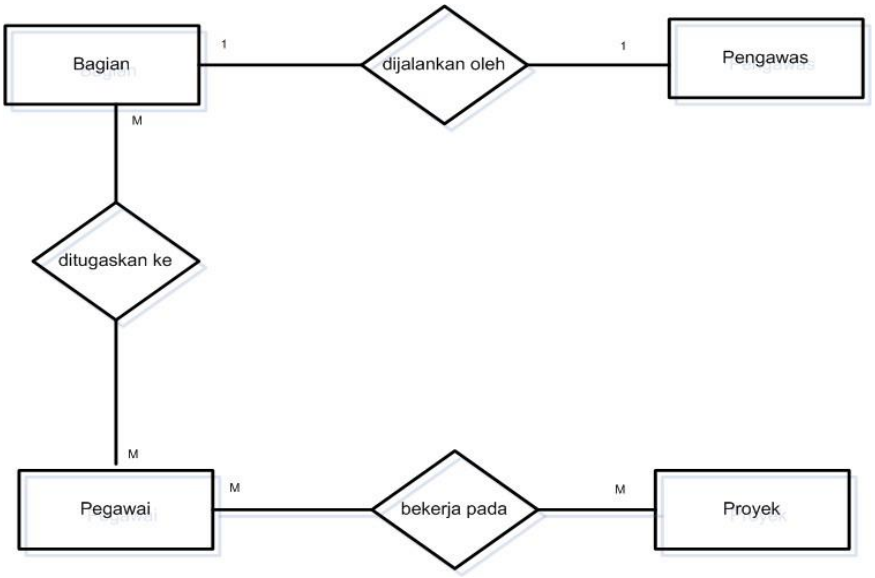


Deskripsi Permasalahan :

- 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



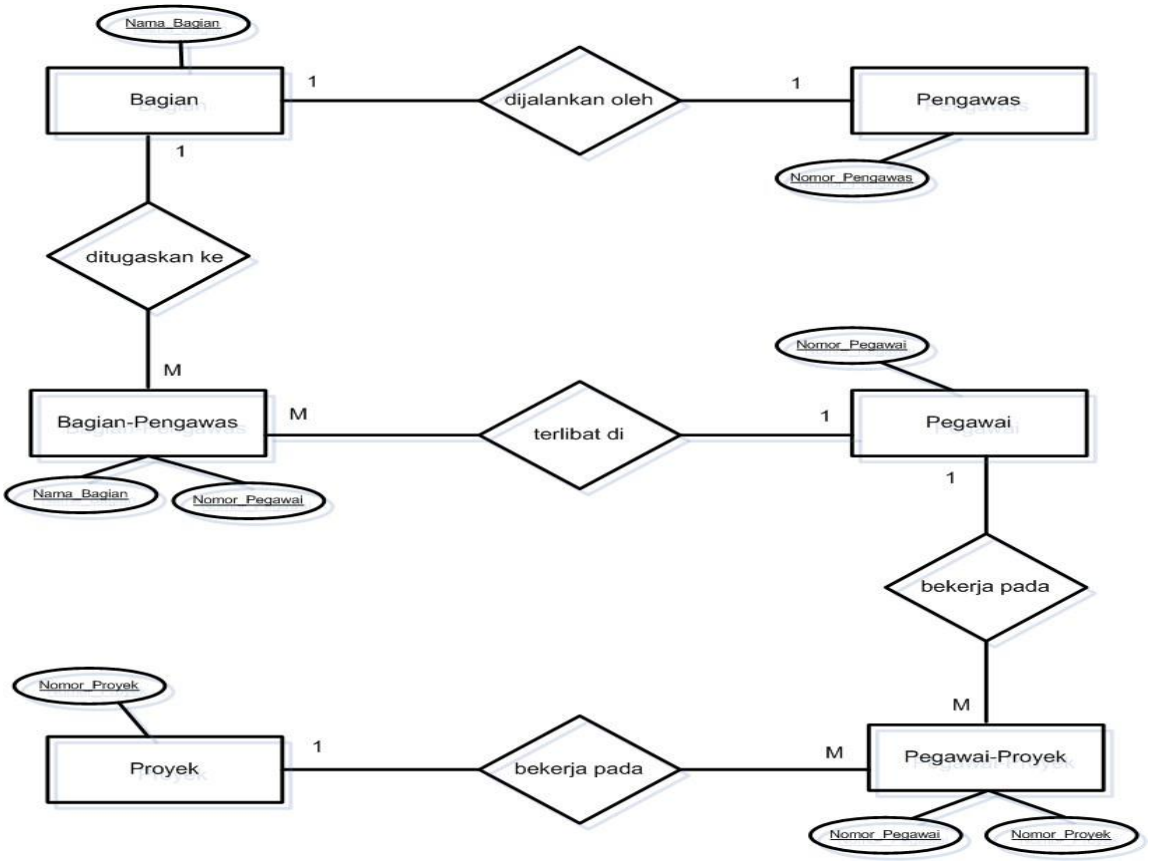
Langkah 5: Menentukan Kunci Utama

Kunci Utama : **Nama Bagian, Nomor Pengawas, Nomor Pegawai, Nomor Proyek. Langkah 6: Menggambarkan ERD berdasarkan kunci**

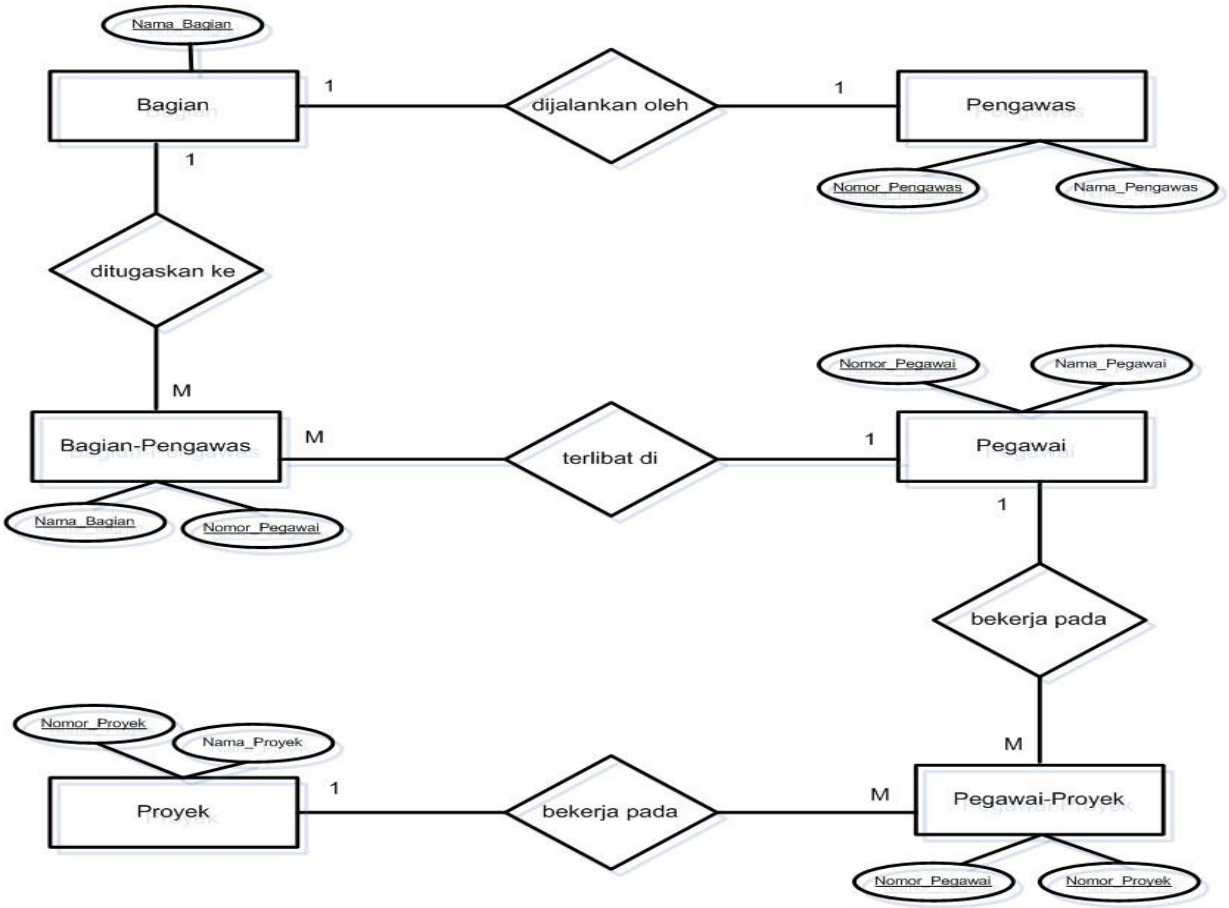
Karena ada dua relasi many-to-many pada ERD sementara, yaitu antara Bagian dan

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

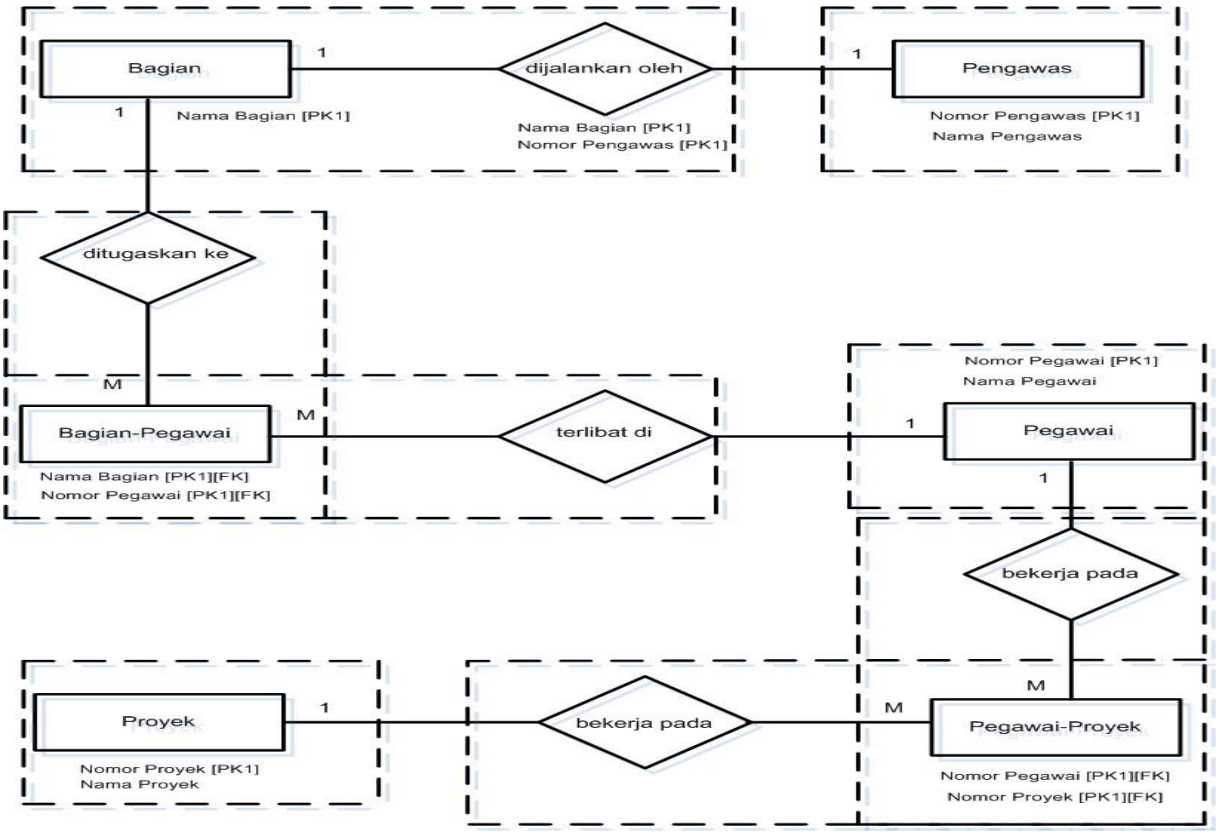
Penggambaran ERD Berdasarkan Kunci


Nama_Bagian
Bagian
1
dijalankan oleh
1
Pengawas
Nomor_Pengawas
1
ditugaskan ke
M
Bagian-Pengawas
M
terlibat di
1
Pegawai
Nomor_Pegawai
Nama_Bagian
Nomor_Pegawai
1
bekerja pada
M
Nomor_Proyek
Proyek
1
bekerja pada
M
Pegawai-Proyek
Nomor_Pegawai
Nomor_Proyek

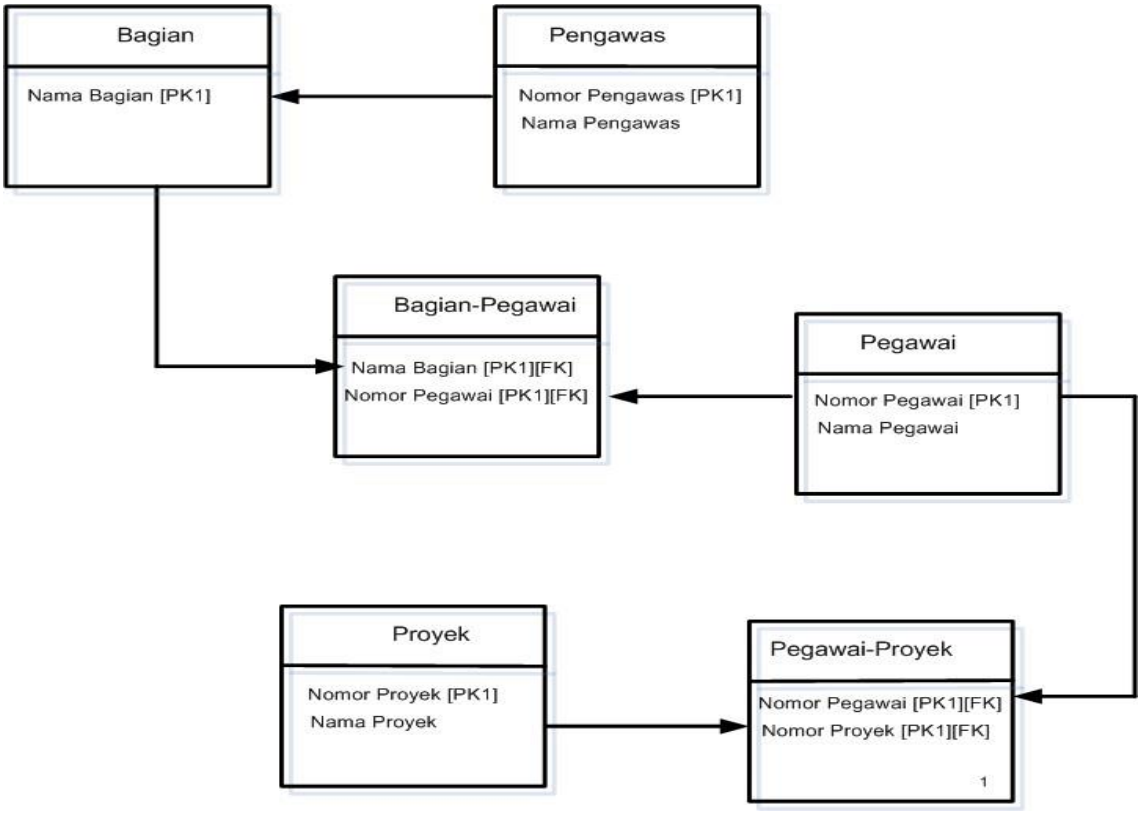
Langkah 7: Menentukan Atribut yang diperlukan



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

A. Definisi Normalisasi

Perancangan basis data seringkali diasosiasikan dengan pembuatan model Entity Relationship (model ER), dimana kelompok-kelompok data dan relasi antar kelompok data tersebut diwujudkan dalam bentuk diagram. Hal itu tidak salah, karena model memang merupakan representasi nyata dalam sebuah perancangaa. Menurut (Fathansyah, 2012), Normalisasi merupakan cara pendekatan lain dalam membangun desain lojik basis data relasional yang tidak secara langsung berkaitan dengan model data, tetapi dengan menerapkan sejumlah aturan dan kriteria standar untuk menghasilkan struktur tabel yang normal. Namun demikian, dalam pelaksanaannya desain lojik basis data relasional didasari oleh transformasi secara hati-hati dari model ER ke bentuk fisik akan menghasilkan hasil yang mirip.

Menurut (Ladjamudin, 2004), menyampaikan beberapa definisi normalisasi, sebagai berikut :

1. Normalisasi adalah suatu proses memperbaiki/membangun dengan model data relasional, dan secara umum lebih tepat dikoneksikan dengan model data logika.
2. Normalisasi adalah proses pengelompokkan data kedalam bentuk tabel atau relasi atau file untuk menyatakan entitas dan hubungan mereka sehingga terwujud satu bentuk database yang mudah untuk dimodifikasi.
3. Normalisasi dapat berguna dalam menjawab 2 pertanyaan mendasar yaitu : “Apa yang dimaksud dengan desain database logical?” dan “Apa yang dimaksud dengan desain databse fisikal yang baik? What is a physical good logical database design?
4. Normalisasi adalah suatu proses untuk mengidentifikasi tabel kelompok atribut yang memiliki ketergantungan yang sangat tinggi antara satu atribut dengan atribut yang lainnya.
5. Normalisasi bisa disebut juga sebagai proses pengelompokkan atribut-atribut dari suatu relasi sehingga membnetuk WELL STRUCTURED RELATION. WELL STRUCTURED 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.

B. Bentuk-bentuk Normalisasi

Bentuk Normalisasi Bentuk normal adalah suatu aturan yang dikenakan pada relasi-relasi dalam basis data, berikut beberapa level yang biasa digunakan pada normalisasi adalah:

1. Bentuk normal pertama (1NF)
2. Bentuk normal kedua (2NF)
3. Bentuk normal ketiga (3NF)
4. Bentuk normal Boyce-Codd (BCNF)
5. Bentuk normal keepat (4NF)
6. Bentuk Normal kelima (5NF)

C. Langkah-Langkah Pembuatan Normalisasi

1. Bentuk Normal Pertama (First Normal Form / 1NF)
   Pada tahap ini dilakukan penghilangan grup elemen yang berulang agar menjadi satu harga tunggal yang berinteraksi diantara setiap baris pada suatu tabel, dan setiap atribut harus mempunyai nilai data yang atomic (atomic value).
2. Bentuk Normal Kedua (2NF)
   Bentuk normal kedua didasari konsep full functional dependency (ketergantungan fungsional sepenuhnya) yang dapat didefinisikan sebagai berikut: Jika A dan B adalah atribut-atribut dari suatu relasi. B dikatakan full functional dependency terhadap A, jika B adalah tergantung fungsional terhadap A, tetapi tidak secara tepat memiliki ketergantungan fungsional dari subset atau himpunan bagian dari A.
3. Bentuk Normal Ketiga (3NF)
   Bentuk normal ketiga adalah suatu relasi dikatakan dalam bentuk normal ketiga (3NF) jika memiliki syarat: berada dalam bentuk normal kedua, setiap atribut bukan kunci haruslah tidak emiliki dependensi transitif (ketergantungan transitif) terhadap kunci primer, dengan kata lain suatu atribut bukan kunci tidak boleh memiliki ketergantungan fungsional terhadap atribut bukan kunci lainnya, selruh atribut bukan kunci pada suatu relasi hanya memiliki ketergantungan fungsional terhadap primary key di relasi itu saja.
4. Bentuk Normal Boyce Codd (BCNF)
   Bentuk BCNF merupakan suatu relasi jika dan hanya jika semua penentu (determinan) adalah kunci kandidat (atribut yang bersifat unik) BCNF merupakan bentuk normal sebagai perbaikan terhadap 3NF. Suatu relasi yang memenuhi BCNF selalu memenuhi 3NF, tetapi tidak untuk sebaliknya.
5. Bentuk Normal Keempat (4NF) 19
   Bentuk normal keempat berkaitan dengan sifat ketergantungan banyak nilai (Multivalued Depedency) pada suatu tabel yang merupakan pengembangan dari ketergantungan fungsional.
6. Bentuk Normal Kelima (PNJF)
   Bentuk normal kelima merupakan nama lain dari Project Join Normal Form (PNJF) yaitu berhubungan dengan ketergantungan relasi antar tabel (Join Dependency)

D. Keuntungan & Syarat Normalisasi Adapun 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

Syarat perlunya normalisasi:

1. Fleksibilitas Struktur database harus menunjang semua cara untuk menampilan data, sehingga ketika user menjalankan aplikasi dan meminta sesuatu dalam database, database harus dapat berajlan memenuhi permintaan user.

2. Integritas data Semua data dalam database yang berkaitan harus terhubung dalam suatu relationship. Sehingga ketika suatu data berubah, maka semua data yang berkaitan dengan data tersebut harus dapat berubah secara otomatis.
3. Efficiency Pada database, ukuran suatu database merupakan hal yang penting. Maka dari itu kita harus mengurangi redudansi data yang bisa menyebabkan ukuran database menjadi besar.
4. Menghindari modification anomaly Desain database yang baik menyajikan suatu keyakinan bahwa ketika user melakukan perubahan dalam database, maka tidak terjadi hal yang tidak diinginkan.

E. Definisi Anomaly

Menurut (Ladjamudin, 2004) anomaly merupakan penyimpangan-penyimpangan atau error atau inkonsistensi data yang terjadi pada saat dilakukan proses insert, delete maupun update dalam suatu basis data.

Jenis Anomaly Terdapat 3 jenis Anomaly (penyimpangan), sebagai berikut:

1. Insertion Anomaly Merupakan error atau kesalahan yang terjadi sebagai akibat operasi insert record/tuple pada sebuah relation. Contoh ada matakuliah baru (CS-600) yang akan diajarkan, maka matakuliah tersebut tidak bisa di insert/disisipkan ke dalam relasi matakuliah sampai ada mahasiswa yang mengambil matakuliah tersebut.
2. Deletion Anomaly Merupakan error atau kesalahan yang terjadi sebagai akibat operasi delete record/tuple pada sebuah relation. Contoh mahasiswa dengan NIM: 0000123 memutuskan untuk batal ikut matakuliah dengan kode CS-400, karena ia merupakan satu-satunya peserta matakuliah tersebut, maka bila record tersebut dihapus akan berakibat hilangnya informasi matakuliah CS400.
3. Update Anomaly Merupakan error atau kesalahan yang terjadi sebagai akibat inkonsistensi data yang terjadi sebagai akibat dari operasi update record/tuple dari sebuah relation. Contoh bila biaya kuliah untuk matakuliah CS-200 akan dinaikkan dari 75 menjadi 100, maka harus dilakukan beberapa kali modifikasi terhadap record-record mahasiswa yang mengambil matakuliah tersebut, agar data tetap konsisten.

Atribut dan Ketergantungan Fungsi Beberapa konsep yang harus diketahui dalam Normalisasi:

1. Field/ Atribut Kunci Key Field / atribut kunci dalam database yaitu Super key, Candidate key, Primary key, Alternate key dan Foreign key.
2. Kebergantungan Fungsi.
    a. Ketergantungan Fungsional (Fungsional Dependent) Keterkaitan antar hubungan antara 2 atribut pada sebuah relasi. Dituliskan dengan cara: A -> B, yang berarti: atribut B fungsionality Dependent terhadap atribut A atau Isi (value) atribut A menentukan isi atribut B.
    b. 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: atribut Y pada relasi R adalah FFD pada atribut 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).

c. Ketergantungan Partial Sebagian dari kunci dapat digunakan sebagai kunci utama. d) Ketergantungan Transitif Menjadi atribut biasa pada suatu relasi tetapi menjadi kunci pada relasi lain. e) Determinan Suatu atribut (field) atau gabungan atribut dimana beberapa atribut lain bergantung sepenuhnya pada atribut tersebut.

BAB VII
BAHASA QUERY FORMAL

A. Pengertian Bahasa Query Formal

Menurut (Fathansyah, 2012), bahasa query merupakan bahasa yang termasuk dalam kategori bahasa tingkat tinggi (high level language) yang digunakan user untuk mendapatkan informasi atau data dari basis data.

Kelompok Bahasa Query Kelompok Bahasa Query Bahasa query dikelompokkan menjadi dua, yaitu:

1. Bahasa procedural User meminta sistem untuk melakukan serangkain operasi terhadap basis data dalam rangka mendapatkan data atau informasi yang diingunkan.
2. Bahasa non procedural User menunjukkan data atau informasi yang diinginkan tanpa menyatakan suatu cara atau prosedur tertentu untuk memperoleh data atau informasi tersebut. Dua Dasar terbentuknya bahasa query menurut Ladjamudin, 2004:
   a. Aljabar Relasional Aljabar relasional merupakan salah satu bahasa manipulasi untuk database relasional. Aljabar relasional merupakan kumpulan operasi terhadap relasi dimana setiap operasi menggunakan satu atau lebih relasi untuk menghasilkan satu relasi yang baru. Aljabar relasional termasuk dalam kategori bahasa procedural yang menyediakan seperangkat operasi untuk memanipulasi data.
   b. Kalkulus Relasional Kalkulus relasional merupakan bahasa manipulasi teoritis yang non procedural. Kalkulus relasional dilandasi dengan teori predicate calculus yang menggunakan fungsi sebagai suatu ekspresi logic. Predikat adalah suatu fungsi yang dapat mengambil nilai benar atau salah tergantung dari subsitusi nilai argument dari fungsi tersebut. Jadi, bila semua argument dari sebuah fungsi disubsitusi dengan suatu nilai, maka fungsi tersebut menjadi suatu ekspresi yang disebut preposisi, yaitu suatu ekspresi yang hanya bernilai. 23

   Operator Aljabar Relational Operator pada aljabar relational dibagi menjadi 2 kelompok:

   1. Operator dasar untuk fundamental operational Operator dasar terdiri dari : selection, projection, cartesian product, set difference dan union.
   2. Operator tambahan untuk additional operasional Operator tambahan terdiri dari setvintersection, theta join, natural join dan division.

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 ( ) 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

KD_MK="306" V SKS=4 (MATAKULIAH)

b. Projection ( )

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

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 „l".

nid, nama_d, nama_mk, thn_akademik,smt, hari,jam_ke, waktu, kelas (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

kota (mahasiswa) kota (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

nama_mhs(alamat="DEPOK" (MAHASISWA))

Query II : tampilkan nama yang berjenis kelamin perempuan

nama_mhs(j_kel ="PEREMPUAN" (MAHASISWA))

Tampilkan query I minus query II :

nama_mhs(alamat="DEPOK"(MAHASISWA))-nama_mhs(j_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:

A

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

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.

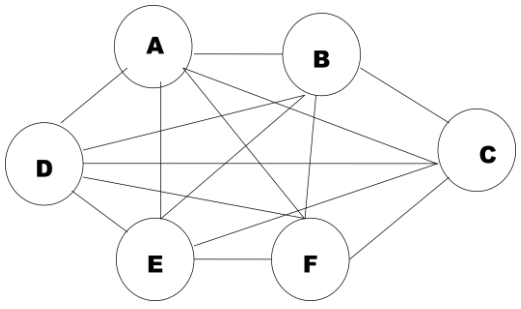
Karakteristik Database terdistribusi, yaitu :

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

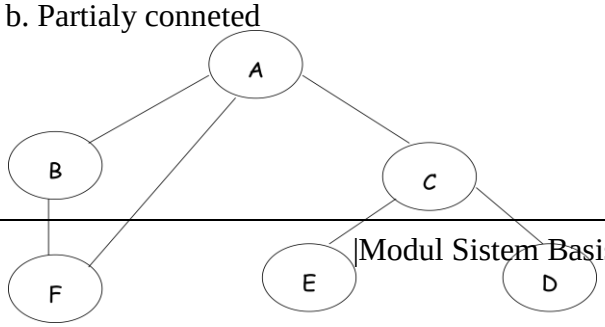
Topologi Distribusi Data

BENTUK-BENTUK TOPOLOGI DISTRIBUSI DATA :

a. Fully Connected network



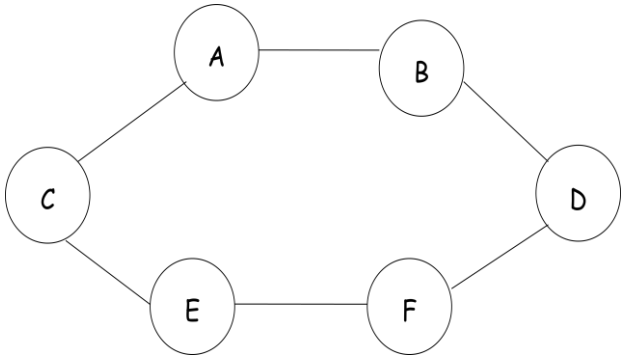
b. Partialy conneted



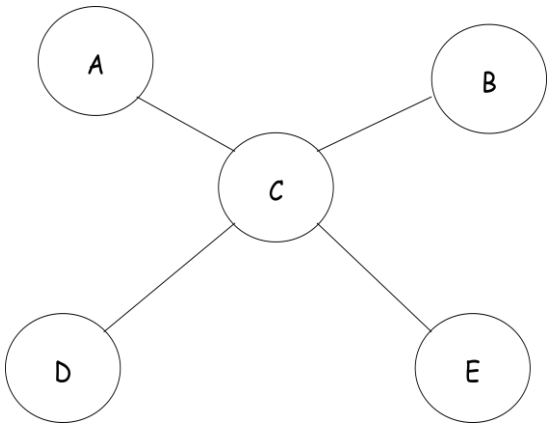
network C. Tree Strutured

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"

   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"

   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"

   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

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

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.

NIM,Nama_Mhs,Mt_Kuliah,Nil_Akhir,Grade,Tuple_ID(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.

NIM,Nama_Mhs,Mt_Kuliah,Nil_Akhir,Grade,Tuple_ID(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

NIM,Nama_Mhs,Mt_Kuliah,Nil_Akhir,Grade,Tuple_ID(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

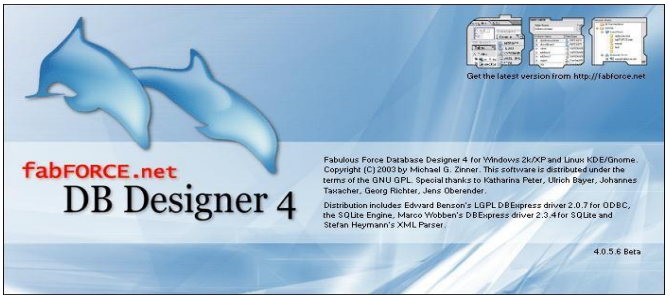
BAB X
Perancangan dan Implementasi Basis Data Menggunakan DB Designer

10.1 Perancangan dan Implementasi Basis Data Menggunakan

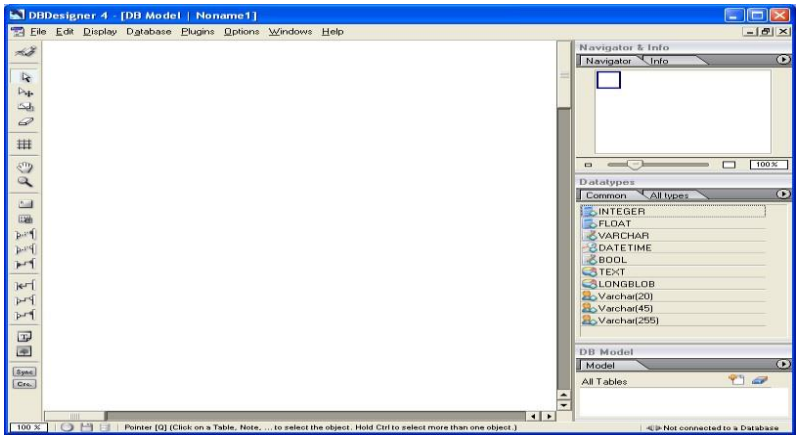
MYSQL Perangkat Lunak Bantu untuk Perancangan Basis Data

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

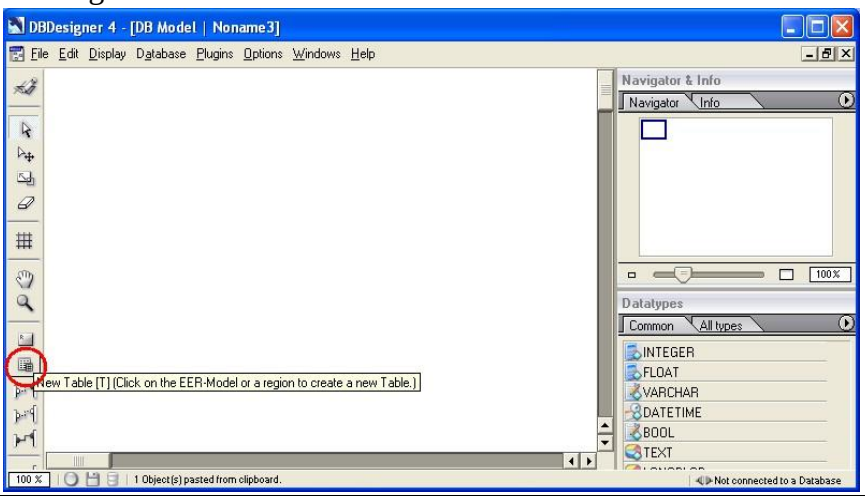
Salah satu perangkat lunak bantu untuk keperluan semacam itu adalah DBDesigner yang dioptimalkan untuk MySQL Database.



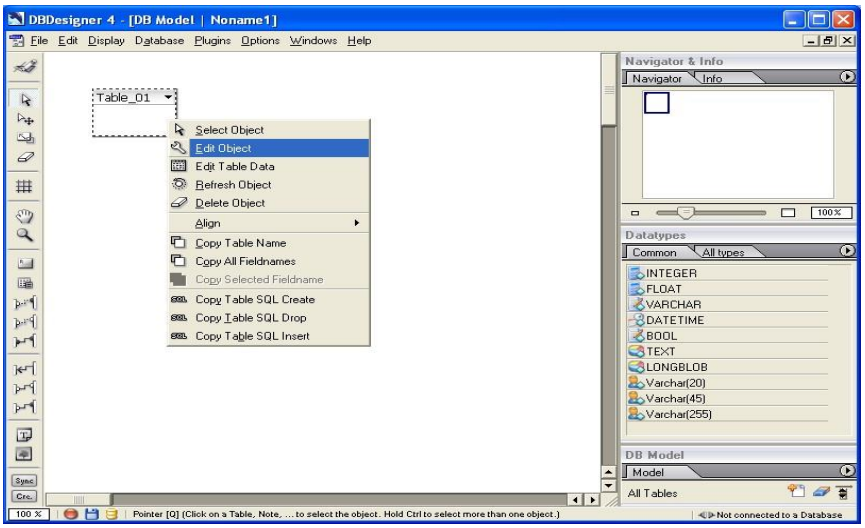
Tampilan jendela DBDesigner.



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.

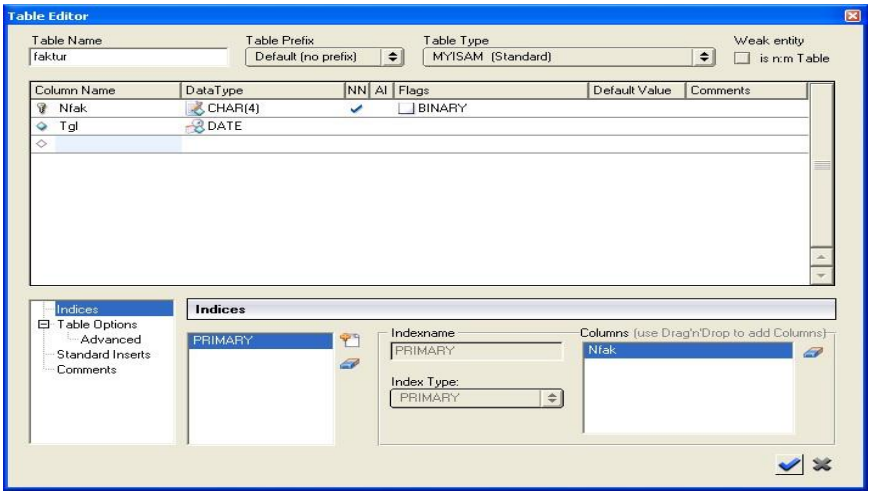


Menu Edit Object akan menampilkan jendela **Table Editor**.

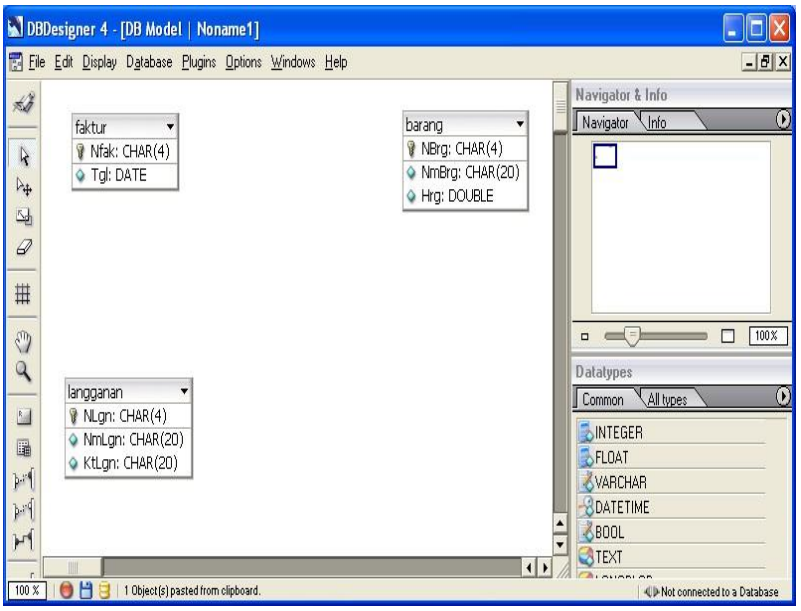
Pada **Table Editor** kita bisa menentukan properties dari tabel seperti nama tabel, tipe data, primary key dsb.

Ubah dan simpanlah properties tabel (Table _01) menjadi tabel **faktur** (struktur tabel

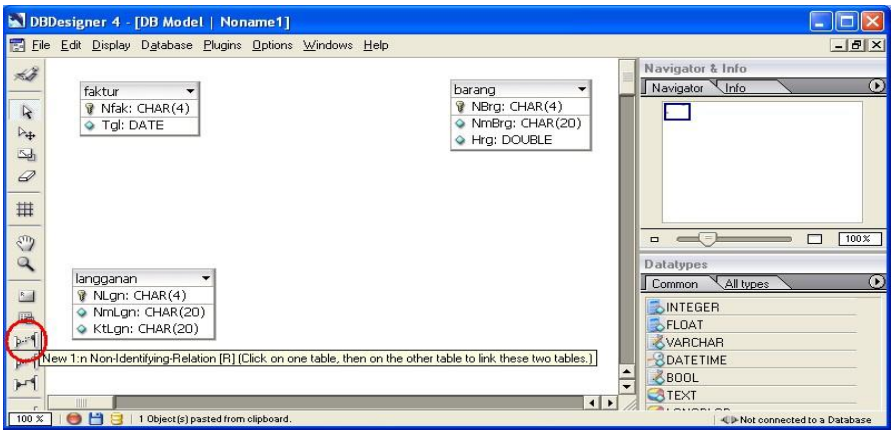
seperti pada pembahasan LRS tanpa ada FK) seperti berikut.



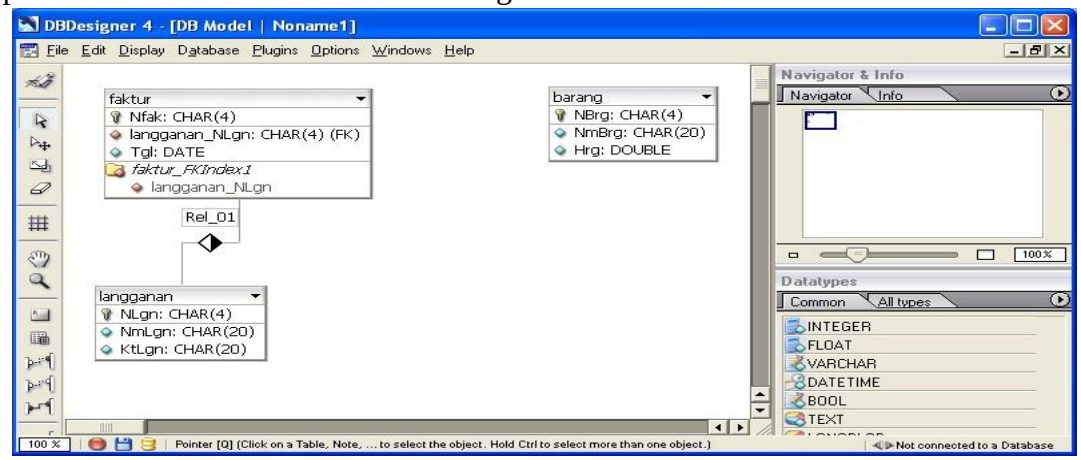
Ulangi langkah-langkah menggunakan komponen **Table** di atas (**tabel faktur**) untuk tabel **barang** dan **langganan** (struktur tabel seperti pada pembahasan LRS tanpa ada FK). Sehingga ada 3 komponen Table seperti gambar berikut



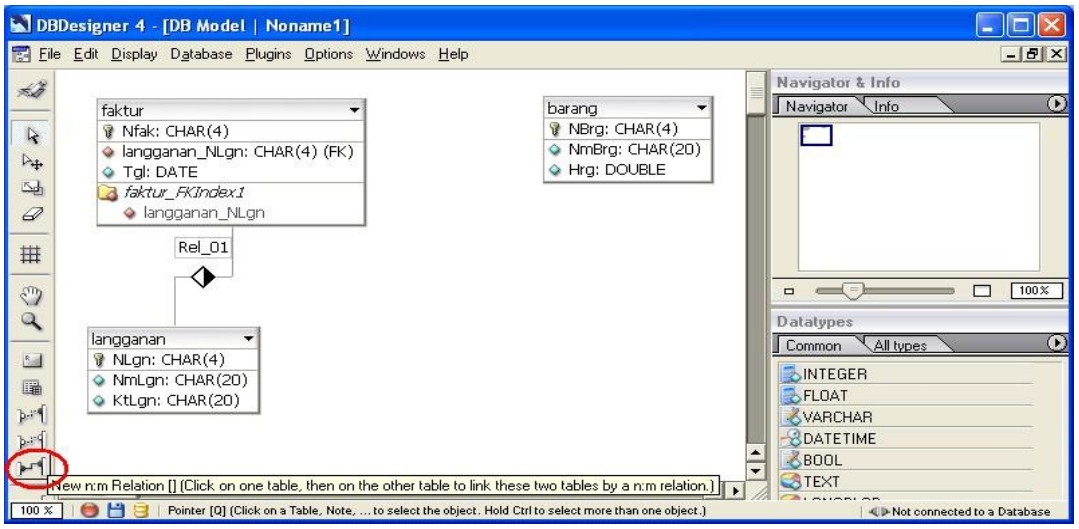
Langkah berikutnya membuat relasi **1-M** antara **langganan** dengan **faktur** dengan cara klik komponen **1-n Relation** pada toolbar seperti di gambar berikut.



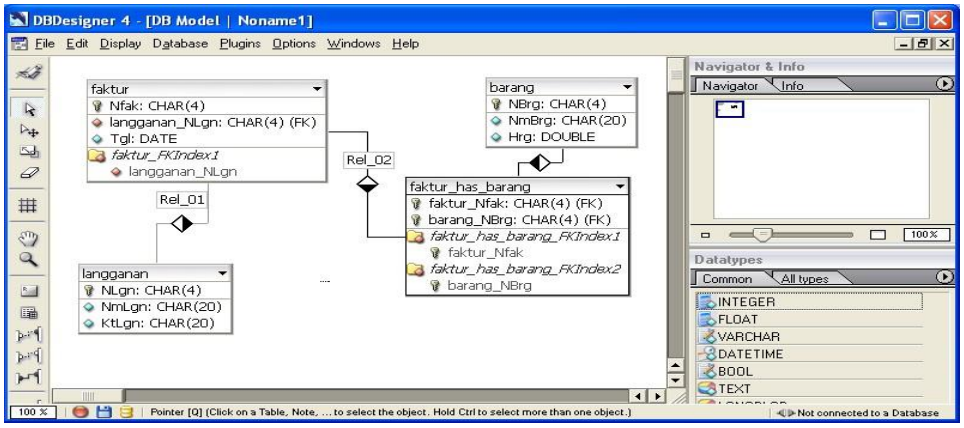
Klik di tabel **langganan** kemudian klik di tabel **faktur**, sehingga muncul komponen relasi yang menghubungkan kedua tabel tsb. dan FK (NLgn) berada pada tabel faktur, seperti gambar berikut



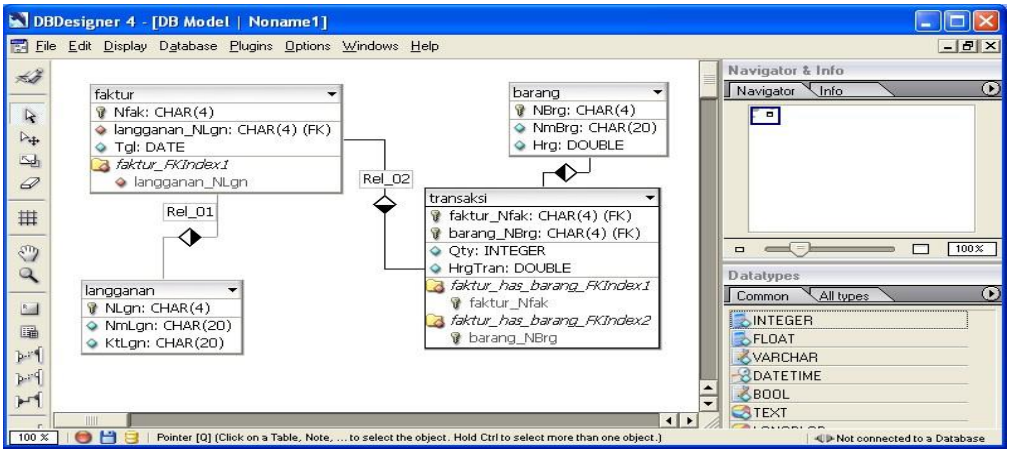
Langkah berikutnya membuat relasi **M-M** antara **faktur** dengan **barang** dengan cara klik komponen **n-m Relation** pada toolbar seperti di gambar berikut



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.

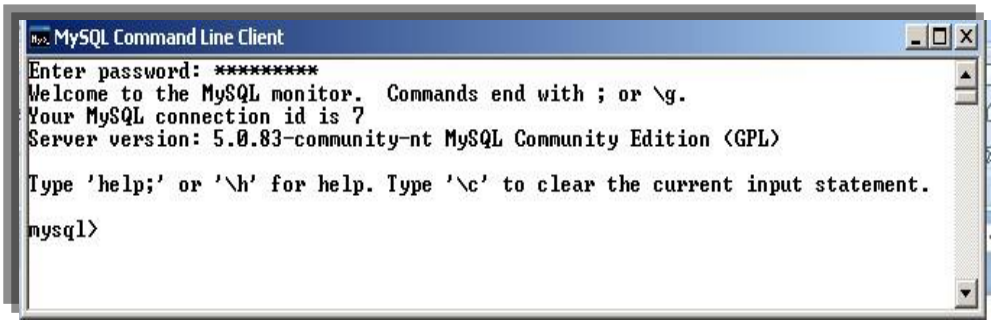


Untuk mengekspor hasil rancangan database ke dalam database digunakan **Database**

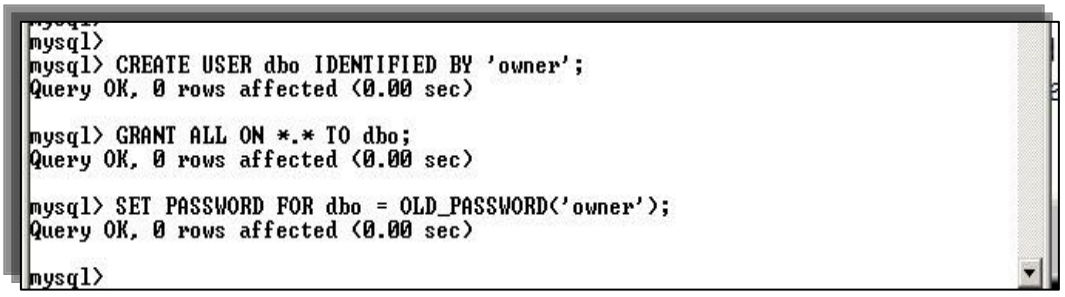
Synchronization. Database yang digunakan pada contoh ini adalah MySQL.

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.



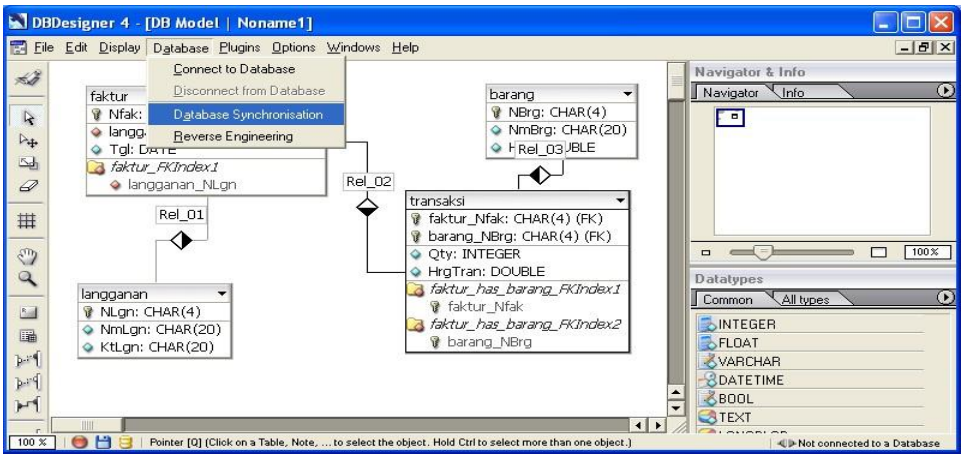
Buat user baru bernama dbo dengan password ”owner”. Ketikkan 3 perintah dibawah ini.



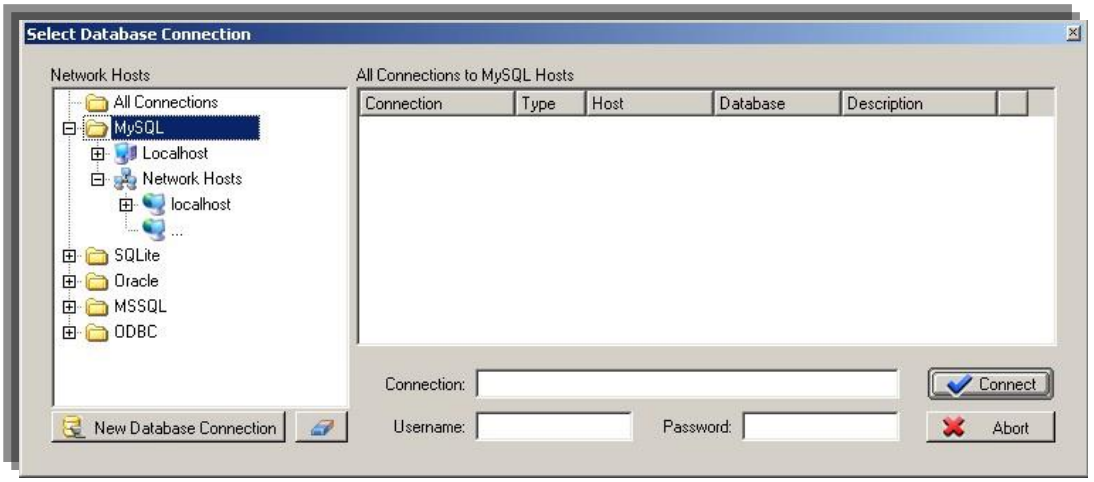
Buat Database baru yaitu dbpenjualan

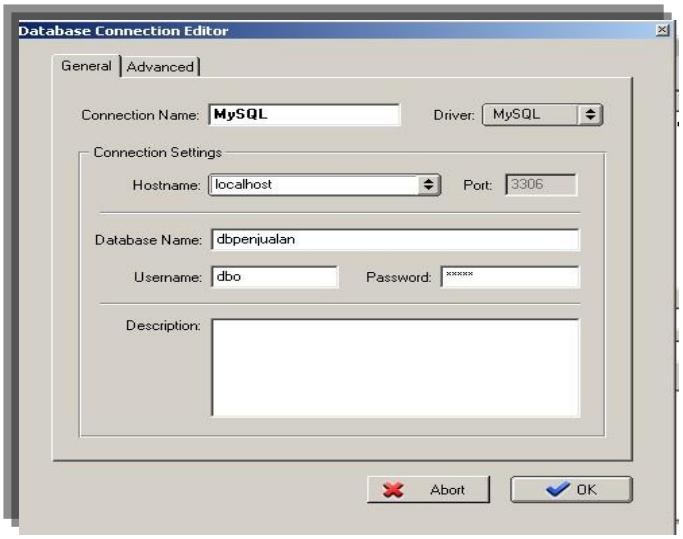
Mengekspor Tabel Hasil Rancangan Ke Server Database

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*

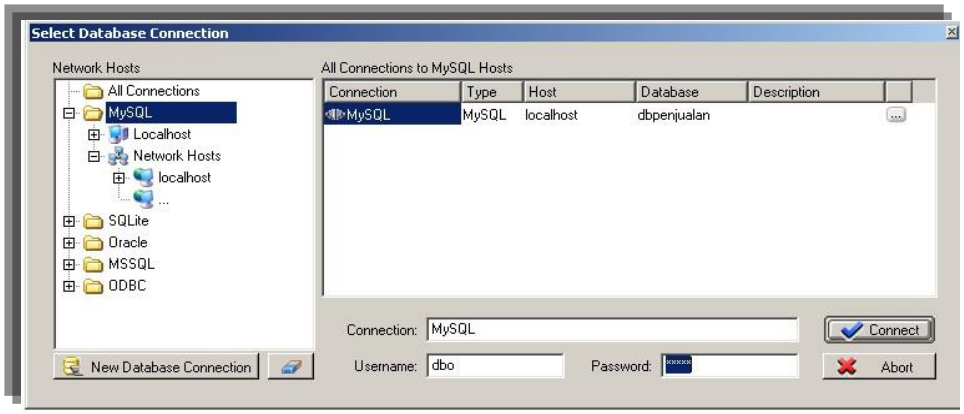
Hostname : *localhost*

Database Name : *dbpenjualan*

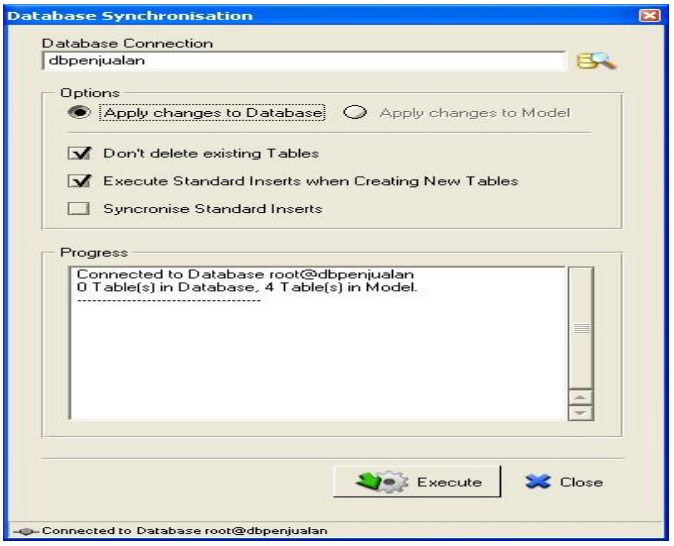
UserName : *dbo*

Password : *owner*

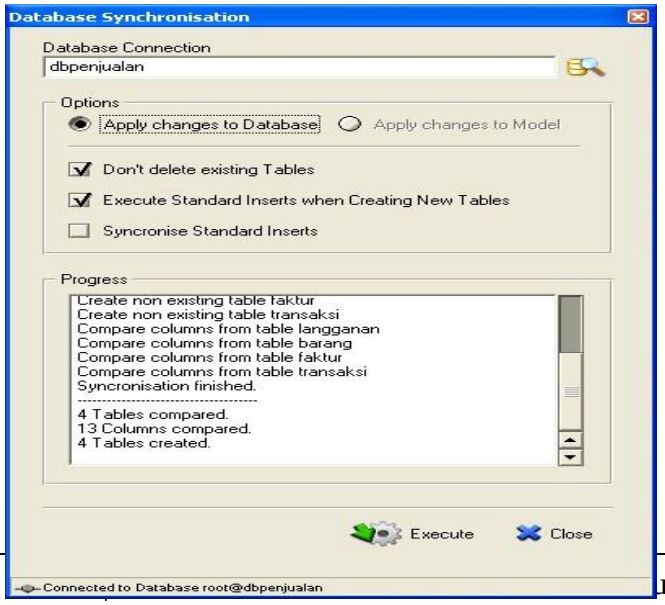
Lalu klikOkKlik **Connect** untuk terkoneksi ke MySQL



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

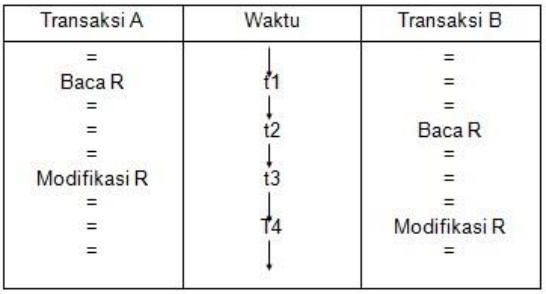
11.1 KONKURENSI

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.



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	t1	Modifkasi R
-	↓	-
-	t2	-
-	↓	-
Modifikasi R	t3	Rollback
-	↓	-

Contoh uncommited Update Problem

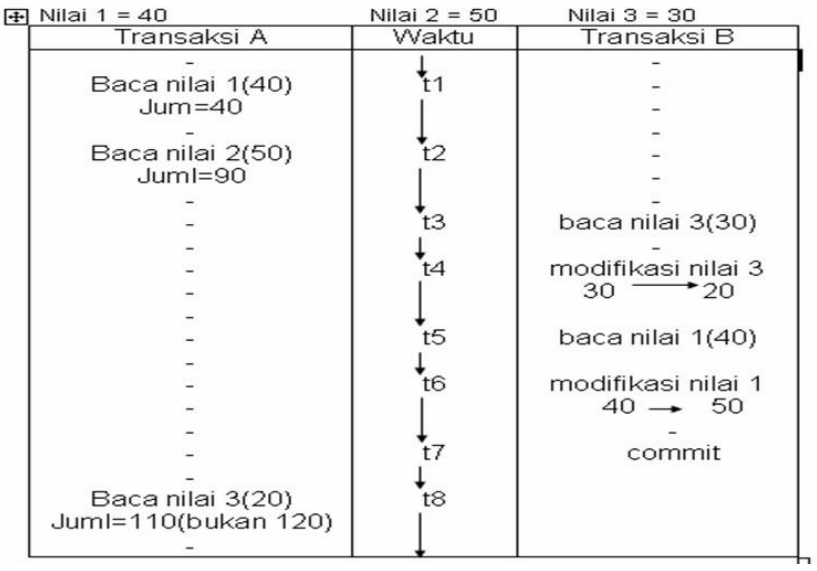
Waktu	Transaksi Simpanan	Transaksi Bunga	Saldo
T1	Read Saldo	.....	1.000.000
T2	Saldo:=saldo+1.000.0000	.....	1.000.000
T3	Write Saldo	.....	2.000.000
T4	.....	Read Saldo	2.000.000
T5	.....	Saldo:= saldo*0.15	2.000.000
T6	.....	Write Saldo	2.300.000
T7	.....	RollBack	2.300.000

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
-	t2	(kunci X)
baca R		-
kunci (S)	t3	-
tunggu		-
:	t4	synchpoint
:		(kunci X dilepas)
tunggu		-
baca R kembali		-
(Kunci S)		-

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

Fathansyah. (2012). Basis Data. Informatika.

Ladjamudin, A. B. Bin. (2004). Konsep Sistem Basis Data dan Implementasinya. Graha Ilmu.

Indrajani. (2009). Sistem Basis Data Dalam Paket Five In One. PT Elex Media Komputindo.

Sukamto, R. A., & Shalahuddin, M. (2018). Rekayasa Perangkat Lunak Terstruktur Dan Berorientasi Objek. Informatika.