BAB II
TINJAUAN PUSTAKA

2.1. Konsep Dasar Sistem

2.2.1. Definisi Sistem

Sistem adalah sebuah tatanan (keterpaduan) yang terdiri atas sejumlah komponen fungsional yang saling berhubungan dan secara bersama-sama bertujuan untuk memenuhi suatu proses tertentu.[7]

Adapun prosedur adalah suatu urut-urutan operasi tulis menulis dan biasanya melibatkan beberapa orang di dalam satu atau lebih departemen yang diterapkan untuk menjamin penanganan yang seragam dari transaksi-transaksi bisnis yang terjadi.[8]

Suatu sistem yang baik harus mempunyai tujuan dan sasaran yang tepat karena hal ini akan sangat menentukan dalam mendefinisikan masukan yang dibutuhkan sistem dan juga keluaran yang dihasilkan.

2.2.2. Elemen Sistem

Elemen-elemen yang terdapat dalam sistem meliputi: [8]

1. Tujuan Sistem

   Tujuan sistem merupakan tujuan dari sistem tersebut dibuat.

2. Batasan Sistem

   Batasan sistem merupakan sesuatu yang membatasi sistem dalam mencapai tujuan sistem. Batasan sistem dapat berupa peraturan-peraturan yang ada dalam organisasi.

3. Kontrol Sistem

   Kontrol atau pengawasan sistem merupakan pengawasan terhadap pelaksanaan pencapaian tujuan dari sistem tersebut. Kontrol sistem dapat berupa kontrol terhadap masukan (*input*), kontrol

terhadap keluaran (*output*), kontrol terhadap pengolahan data dan kontrol terhadap umpan balik.

4. Input

   Input merupakan elemen sistem yang bertugas untuk menerima masukan data.

5. Proses

   Proses merupakan elemen dari sistem yang bertugas untuk mengolah atau memproses seluruh masukan data menjadi suatu informasi yang lebih berguna.

6. Output

   Output merupakan hasil dari input yang telah diproses oleh bagian pengolah dan merupakan tujuan akhir sistem.

7. Umpan Balik

   Umpan balik merupakan elemen dalam sistem yang bertugas mengevaluasi bagian dari output yang dikeluarkan, dimana elemen ini sangat penting demi kemajuan sebuah sistem. Umpan balik ini dapat berupa perbaikan sistem maupun pemeliharaan sistem.

2.2.3. Karakteristik Sistem

Suatu sistem mempunyai karakteristik atau sifat - sifat yang tertentu, yaitu :[7]

1. Komponen-komponen Sistem (Components)

   Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang artinya saling bekerja sama membentuk satu kesatuan. Komponen - komponen sistem atau elemen - elemen sistem dapat berupa suatu subsistem atau bagian - bagian dari sistem. Setiap subsistem mempunyai sifat - sifat dari sistem untuk menjalankan suatu fungsi tertentu dan mempengaruhi proses sistem secara keseluruhan. Suatu sistem dapat mempunyai suatu sistem yang lebih besar yang disebut dengan *supra system.*

Misalnya suatu sistem yang lebih besar dapat disebut dengan suatu sistem dan industri yang merupakan sistem yang lebih besar dapat disebut dengan *supra system*.

2. Batas Sistem (*Boundary*)

   Batas sistem (*boundary*) merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. Batas sistem ini memungkinkan suatu sistem dipandang sebagai satu kesatuan. Batas suatu sistem menunjukkan ruang lingkup (*scope*) dari sistem tersebut.

3. Lingkungan Luar Sistem (*Environments*)

   Lingkungan luar (*Environment*) dari suatu sistem adalah apapun diluar batas dari sistem yang mempengaruhi operasi sistem.

4. Penghubung Sistem (*Interface*)

   Penghubung (*interface*) merupakan media penghubung antara satu subsistem dengan subsitem yang lainnya. Melalui penghubung ini memungkinkan sumber - sumber daya mengalir dari satu subsistem ke subsistem yang lainnya. Keluaran (*output*) dari satu subsistem akan menjadi masukkan (*input*) untuk subsistem yang lainnya dengan melalui penghubung. Dengan penghubung satu subsistem dapat berintegrasi dengan subsistem yang lainnya membentuk satu kesatuan.

5. Masukan Sistem (*Input*)

   Masukan (input) adalah energi yang dimasukkan ke dalam sistem. Masukan dapat berupa masukan perawatan (*maintenance input*) dan masukkan sinyal (*signal input*). *Maintenance input* adalah energi yang dimasukkan supaya sistem tersebut dapat beroperasi. *Signal input* adalah energi yang diproses untuk didapatkan keluaran. Sebagai contoh di dalam sistem komputer, program adalah *maintenance input* yang

digunakan untuk mengoperasikan komputernya dan data adalah *signal input* untuk diolah menjadi informasi.

6. Keluaran Sistem (*Output*)

   Keluaran (o*utput*) adalah hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan sisa pembuangan. Keluaran dapat merupakan masukan untuk subsistem yang lain atau kepada suprasistem.

7. Pengolah (*Process*)

   Suatu sistem dapat mempunyai suatu bagian pengolah yang akan merubah masukkan menjadi keluaran.

8. Sasaran (Objectives) atau Tujuan (*Goal*)

   Suatu sistem pasti mempunyai tujuan (*goal*) atau sasaran (*objective*). Kalau suatu sistem tidak mempunyai sasaran, maka operasi sistem tidak akan ada gunanya. Suatu sistem dikatakan berhasil bila mengenai sasaran atau tujuannya.

2.2.4. Klasifikasi Sistem

Sistem dapat diklasifikasikan dari beberapa sudut pandang, diantaranya sebagai berikut ini: [7]

1. Sistem diklasifikasikan sebagai sistem abstrak (*abstract system*) dan sistem fisik (*physical system*).

   Sistem abstrak adalah sistem yang berupa pemikiran atau ide-ide yang tidak tampak secara fisik. Misalnya sistem teologia, yaitu sistem yang berupa pemikiran – pemikiran hubungan antara manusia dengan Tuhan. Sistem fisik merupakan sistem yang ada secara fisik. Misalnya sistem komputer, sistem akuntansi, sistem produksi dan lain sebagainya.

2. Sistem diklasifikasikan sebagai sistem alamiah (*natural system*) dan sistem buatan manusia (*human made system*).

   Sistem alamiah adalah sistem yang terjadi melalui proses alam, tidak dibuat manusia. Misalnya sistem perputaran bumi.

Sistem buatan manusia adalah sistem yang dirancang oleh manusia. Sistem buatan manusia yang melibatkan interaksi antara manusia dengan mesin disebut dengan *human-machine system* atau ada yang menyebut dengan *man-machine system*. Sistem informasi merupakan contoh *man-machine system*, karena menyangkut penggunaan komputer yang berinteraksi dengan manusia.

3. Sistem diklasifikasikan sebagai sistem tertentu (*deterministic system*) dan sistem tak tentu (*probabilistic system*).

   Sistem tertentu beroperasi dengan tingkah laku yang sudah dapat diprediksi. Interaksi diantara bagian-bagiannya dapat dideteksi dengan pasti, sehingga keluaran dari sistem dapat diramalkan. Sistem komputer adalah contoh dari sistem tertentu yang tingkah lakunya dapat dipastikan berdasarkan program – program yang dijalankan. Sistem tak tentu adalah sistem yang kondisi masa depannya tidak dapat diprediksi karena mengandung unsur probabilitas.

4. Sistem diklasifikasikan sebagai sistem tertutup (*closed system*) dan sistem terbuka (*open system*).

   Sistem tertutup merupakan sistem yang tidak berhubungan dan tidak terpengaruh dengan lingkungan luarnya. Sistem ini bekerja secara otomatis tanpa adanya turut campur tangan dari pihak diluarnya. Secara teoritis sistem tertutup ini ada, tetapi kenyataannya tidak ada sistem yang benar-benar tertutup, yang ada hanyalah *relatively closed system* (secara relatif tertutup, tidak benar-benar tertutup). Sistem terbuka adalah sistem yang berhubungan dan terpengaruh dengan lingkungan luarnya. Sistem ini menerima masukan dan menghasilkan keluaran untuk lingkungan luar atau subsistem yang lainnya. Karena sistem sifatnya terbuka dan terpengaruh oleh lingkungan luarnya, maka suatu sistem harus mempunyai suatu sistem pengendalian yang

baik. Sistem yang baik harus dirancang sedemikian rupa, sehingga secara relatif tertutup karena sistem tertutup akan bekerja secara otomatis dan terbuka hanya untuk pengaruh yang baik saja.

2.2.5. Kriteria Sistem Yang Baik

Kriteria sistem yang baik antara lain: [7]

1. Kegunaan

   Sistem harus dapat menghasilkan informasi yang tepat waktu dan relevan untuk proses pengambilan keputusan.

2. Ekonomis

   Sistem harus dapat menyumbang sesuai nilai tambah sekurang-kurangnya sebesar biayanya.

3. Keandalan

   Keluaran dari sistem harus mempunyai tingkat ketelitian yang tinggi dan dapat beroperasi secara efektif dan lebih sempurna.

4. Kapasitas

   Sistem harus cukup sederhana sehingga struktur dan operasinya dapat dengan mudah dipahami dan prosedur mudah diikuti.

5. Fleksibilitas

   Sistem harus cukup fleksibel untuk menampung perubahan.

2.2. Konsep Dasar Basis Data

2.2.1. Pengertian Basis Data

Basis data terdiri atas dua kata, yaitu Basis dan Data. Basis dapat diartikan sebagai markas atau gudang tempat bersarang atau berkumpul. Sedangkan data adalah representasi fakta dunia nyata yang mewakili suatu objek seperti manusia (pegawai, siswa, pembeli, pelanggan), barang, hewan, peristiwa, konsep, keadaan dan lain sebagainya yang direkam dalam bentuk angka, huruf, simbol, teks, gambar, bunyi atau kombinasinya. Jadi basis data adalah

kumpulan file atau arsip yang saling berhubungan yang disimpan dalam media penyimpanan elektronis.[6]

Basis data (bahasa Inggris: database), atau sering pula dieja basisdata, adalah kumpulan informasi yang disimpan di dalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data tersebut. Perangkat lunak yang digunakan untuk mengelolah dan memanggil basis data disebut sistem manajemen basis data database management sistem, (DBMS).[3]

Basis data sendiri dapat didefinisikan dalam sejumlah sudut pandang seperti: [6]

1. Himpunan kelompok data (arsip) yang saling berhubungan yang diorganisasikan sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah.
2. Kumpulan data yang saling berhubungan yang disimpan secara bersama dan tanpa pengulangan (*redudansi*) yang tidak perlu, untuk memenuhi berbagai kebutuhan.
3. Kumpulan file, tabel, arsip yang saling berhubungan yang disimpan dalam media penyimpanan elektronis

2.2.2. Hirarki Data

Secara tradisional data diorganisasikan ke dalam suatu hirarki yang terdiri atas elemen data, record dan berkas. [7]

1. Entity

   Merupakan orang, tempat kejadian atau konsep yang informasinya direkam. Dalam hal rawat inap misalnya, entity adalah puskesmas, petugas, pasien dan obat.

2. Atribut

   Setiap entity mempunyai atribut atau sebutan untuk mewakili suatu entity. Sebuah obat dapat dilihat atributnya, misalnya kode obat,

nama obat dan jenis obat. Atribut juga disebut data elemen, data field dan item data.

3. Data value

   Merupakan data aktual atau informasi yang disimpan pada tiap elemen data atau atribut. Atribut nama karyawan disimpan, sedangkan data value adalah “agus” yang merupakan isi dari data nama petugas tersebut.

4. Record

   Merupakan gabungan atau kumpulan elemen yang saling berkaitan memberikan informasi tentang entity secara lengkap. Satu record akan mewakili satu data atau informasi tentang seseorang atau objek lainnya, misalnya kode obat, nama obat, jenis obat, jumlah obat dan harga obat.

5. File

   Merupakan kumpulan dari record data yang berkaitan dengan subjek data.

2.2.3. Model Data

Model basis data menyatakan hubungan antar rekaman yang tersimpan dalam basis data. Beberapa literatur menggunakan istilah struktur data logis untuk menyatakan keadaan ini. Model dasar yang paling umum ada 3 macam, yaitu: [13]

1. Model Hirarkis

   Model hirarkis biasa disebut model pohon, karena menyerupai pohon yang dibalik. Model ini menggunakan pola hubungan orang tua – anak. Setiap simpul (biasa dinyatakan dengan lingkaran atau kotak) menyatakan sekumpulan medan. Simpul yang terhubung ke simpul pada level di bawahnya disebut orang tua. Setiap orang tua bisa memiliki satu (hubungan 1:1) atau beberapa anak (hubungan 1:M), tetapi setiap anak hanya memiliki satu orang tua. Simpul – simpul yang dibawahi

oleh simpul orang tua disebut anak. Simpul orang tua yang tidak memiliki orang tua disebut akar. Simpul yang tidak mempunyai anak disebut daun. Adapun hubungan antara anak dan orang tua disebut cabang.

2. Model Jaringan

Model jaringan distandarisasi pada tahun 1971 oleh *Data Base Task Group* (DBTG). Itulah sebabnya disebut model DBTG. Model ini juga disebut model CODASYL (Conference *On Data Sistem Languages*), karena DBTG adalah bagian dari CODASYL. Model ini menyerupai model hirarkis, dengan perbedaan suatu simpul anak bisa memiliki lebih dari satu orang tua. Oleh karena sifatnya demikian, model ini bisa menyatakan hubungan 1**:**1 (satu orang tua punya satu anak), 1**:**M (satu orang tua punya banyak anak), maupun N**:**M (beberapa anak bisa mempunyai beberapa orang tua). Pada model jaringan, orang tua disebut pemilik dan anak disebut anggota.

3. Model Relasional

Model relasional merupakan model yang paling sederhana sehingga mudah digunakan dan dipahami oleh pengguna, serta merupakan yang paling populer saat ini. Model ini menggunakan sekumpulan tabel berdimensi dua yang disebut relasi atau tabel dengan masing-masing relasi tersusun atas tupel atau baris dan atribut. Relasi dirancang sedemikian rupa sehingga dapat menghilangkan kemubaziran dan menggunakan kunci tamu untuk untuk berhubungan dengan relasi lain. DBMS (*Database Management System*)yang bermodelkan relasional biasa disebut RDBMS (*Relational Database Management System*).

2.2.4. Operasi Dasar Basis Data

Di dalam sebuah *disk*, basis data dapat diciptakan dan dapat pula ditiadakan. Dalam sebuah *disk* dapat pula menempatkan beberapa basis data. Sementara dalam sebuah basis data dapat menempatkan satu atau lebih file atau tabel. Pada file atau tabel inilah sesungguhnya data disimpan dan ditempatkan. Setiap basis umumnya dibuat untuk mewakili semesta data yang spesifik, misalnya basis data kepegawaian, basis data akademik, basis data pergudangan (*inventory*) dan lain sebagainya. Sementara dalam basis data *inventory* atau pergudangan, kita dapat menempatkan tabel perusahaan, tabel supplier, tabel pelanggan dan tabel barang, dan seterusnya. Karena itu, operasi-operasi dasar yang dapat dilakukan berkenaan dengan basis data meliputi: [10]

1. Pembuatan basis data baru (*create database*), yang identik dengan pembuatan lemari arsip yang baru.
2. Penghapusan basis data (*drop database*), yang identik dengan perusakan lemari arsip sekaligus beserta isinya jika ada.
3. Pembuatan tabel baru ke dalam suatu basis data (*create table*), yang identik dengan penambahan map arsip baru ke dalam lemari arsip yang telah ada.
4. Penghapusan tabel dari suatu basis data (*drop table*), yang identik dengan perusakan map arsip lama yang ada di sebuah lemari arsip.
5. Penambahan atau pengisian data baru ke sebuah tabel di dalam sebuah basis data (*insert*), yang identik dengan penambahan lembaran arsip ke dalam sebuah map arsip.
6. Pengambilan data dari sebuah tabel (*retrieve / search*), yang identik dengan pencarian lembaran arsip dari sebuah map arsip.
7. Pengubahan data dari sebuah tabel (*update*), yang identik dengan perbaikan isi lembaran arsip yang ada disebuah map arsip.
8. Penghapusan data dari sebuah tabel (*delete*), yang identik dengan penghapusan sebuah lembaran arsip yang ada disebuah map arsip.

Operasi yang berkenaan dengan pembuatan objek, yaitu basis data dan tabel merupakan operasi awal yang hanya dilakukan sekali dan berlaku untuk seterusnya. Sedangkan operasi yang berkenaan dengan isi tabel, yaitu data merupakan operasi rutin yang akan berlangsung secara berulang – ulang sehingga operasi ini lebih tepat mewakili aktivitas pengelolaan (*management*) dan pengolahan (*processing*) data dalam sebuah basis data.

2.2.5. Keuntungan Basis Data

Penyusunan suatu basis data digunakan untuk mengatasi permasalahan-permasalahan pada saat pengolahan basis data. Basis data dapat dikembangkan dengan benar sesuai dengan batasan/kaidahnya, basis data akan memberikan beberapa keuntungan yaitu [10] :

a. Kerangkapan data dapat diminimalkan.
b. Inkonsistensi data dapat dihindari.
c. Data dalam basis data dapat digunakan secara bersama (*multi-user*).
d. Standarisasi data dapat dilakukan.
e. Pembatasan untuk keamanan data dapat diterapkan
f. Integritas data dapat terpelihara
g. Perbedaan kebutuhan data dsapat diseimbangkan.

2.3. Analisa Sistem

2.3.1. Definisi Analisis Sistem

Analisis sistem adalah penguraian dari suatu sistem informasi yang utuh kedalam bagian-bagian komponennya dengan maksud untuk mengidentifikasi dan mengevaluasi permasalahan - permasalahan, kesempatan, hambatan yang terjadi dan kebutuhan

yang diharapkan sehingga dapat diusulkan perbaikan- perbaikannya. [7]

2.3.2. Langkah – Langkah Analisis Sistem

Dalam tahap analisis sistem terdapat langkah – langkah dasar yang harus dilakukan oleh analis sistem sebagai berikut : [7]

1. Mengidentifikasi Masalah

   Mengidentifikasi masalah merupakan langkah pertama yang dilakukan dalam tahap analisis sistem. Tugas-tugas yang harus dilakukan dalam tahap ini adalah :

   a. Mengidentifikasi penyebab masalah.
   b. Mengidentifikasi titik keputusan.
   c. Mengidentifikasikan personil-personil kunci.

2. Memahami Kerja dari Sistem yang Ada

   Langkah ini dilakukan dengan mempelajari secara terinci bagaimana sistem yang ada beroperasi. Untuk itu diperlukan data yang diperoleh melalui penelitian, dengan langkah – langkah sebagai berikut :

   a. Menentukan jenis penelitian
   b. Merencanakan jadwal penelitian
   c. Membuat penugasan penelitian
   d. Membuat agenda wawancara
   e. Mengumpulkan hasil penelitian

3. Menganalisis hasil penelitian

   Langkah ini dilakukan berdasarkan data yang telah diperoleh dari hasil penelitian yang sudah dilakukan. Menganalisis sistem dibagi menjadi dua, yaitu menganalisa berdasarkan prinsip penelitian dan menganalisis berdasarkan pokok – pokok analisis.

4. Membuat laporan hasil analis

   Mengambil keputusan dari hasil penelitian yang telah berjalan. Tujuan dari pembuatan laporan adalah :

a. Laporan bahwa analisis telah dilakukan.
b. Meluruskan kesalahpahaman mengenai analisa yang ditemukan dan dianalisis oleh analis sistem tetapi tidak sesuai menurut manajemen.
c. Meminta pendapat dan saran dari pihak manajemen.
d. Meminta pihak manajemen untuk melakukan tindakan selanjutnya (dapat berupa meneruskan ke tahap desain sistem atau menggantikan proyek bila dipandang tidak layak).

2.3.3. Tujuan Analisis Sistem

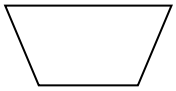
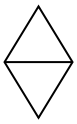
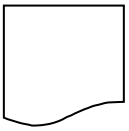
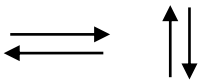
1. Membuat keputusan apabila sistem ini mempunyai masalah atau sudah tidak berfungsi secara baik dan hasil analisisnya digunakan sebagai dasar untuk memperbaiki sistem.
2. Mengetahui ruang lingkup pekerjaan yang akan ditangani.
3. Memahami sistem yang sedang berjalan saat ini.
4. Mengidentifikasikan masalah dalam mencari solusinya.

2.3.4. Alat Bantu Analisis Sistem

Alat Bantu Analisis sistem yang digunakan yaitu: Bagan Alir Dokumen (*Document flowchart)*. Merupakan bagan alir yang menunjukkan arus dari laporan dan formulir termasuk tembusan-tembusannya. Bagan alir dokumen ini menggunakan simbol-simbol yang sama dengan yang digunakan di dalam bagan alir sistem. [7]

Simbol - simbol untuk menyajikan bagan alir dokumen (*Document flowchart*) sebagai berikut :

Simbol	Keterangan
	DOKUMEN Menunjukan dokumen input dan output baik untuk proses manual atau komputer

	KEGIATAN MANUAL Menunjukkan pekerjaan manual
	SIMPANAN Menunjukkan pengarsipan file
	PENGURUTAN OFFLINE Menunjukkan proses pengurutan data di luar proses komputer
	PITA KONTROL Menunjukkan penggunaan pita kontrol dalam *batch control total* untuk pencocokan dalam proses *batch processing*
	GARIS ALIH Simbol garis alir menunjukkan arus dari proses
	PENGHUBUNG Menunjukkan penghubung ke halaman yang masih sama atau ke halaman lain

Tabel 2.1 : Simbol Sistem Prosedur Diagram

Sumber : [7]

2.4. Tahap Desain Sistem (Perancangan Sistem)

2.4.1. Definisi Perancangan Sistem

Dari sekian banyak yang memberikan pengertian mengenai perancangan sistem, akhirnya perancangan sistem dapat diartikan sebagai berikut : [7]

1. Tahap setelah analisis dari siklus pengembangan sistem.
2. Pendefinisian dari kebutuhan-kebutuhan fungsional.
3. Persiapan untuk rancang bangun implentasi.
4. Menggambarkan bagaimana suatu sistem dibentuk.

5. Sistem dibentuk dapat berupa penggambaran, perencanaan dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah ke dalam satu kesatuan yang utuh dan berfungsi.
6. Termasuk menyangkut mengkonfigurasi dari komponen perangkat lunak dan perangkat keras dari suatu sistem.

2.4.2. Tujuan Perancangan Sistem

Tahap perancangan sistem mempunyai dua maksud atau tujuan utama, yaitu sebagai berikut :

1. Untuk memenuhi kebutuhan kepada pemakai sistem.
2. Untuk memberikan gambaran yang jelas dan rancang bangun yang lengkap kepada pemrogram komputer dan ahli – ahli teknik lainnya yang terlibat.

Tujuan kedua ini lebih condong pada desain sistem yang terinci, yaitu pembuatan rancang bangun yang jelas dan lengkap untuk nantinya digunakan untuk pembuatan progam komputernya. Untuk mencapai tujuan ini, analis sistem harus dapat mencapai sasaran-sasaran sebagai berikut:

1. Desain sistem harus berguna, mudah dipahami dan nantinya mudah digunakan. Ini berarti bahwa data harus mudah ditangkap, metodenya harus mudah diterapkan dan informasi harus mudah dihasilkan serta mudah dipahami dan digunakan.
2. Desain sistem harus dapat mendukung tujuan utama perusahaan sesuai dengan yang didefinisikan pada tahap perencanaan sistem yang dilanjutkan pada tahap analisis sistem.
3. Desain sistem harus efisien dan efektif untuk dapat mendukung pengolahan transaksi, pelaporan manajemen dan mendukung keputusan yang akan dilakukan oleh manajemen, termasuk tugas - tugas yang lainnya yang tidak dilakukan oleh komputer.
4. Desain sistem harus dapat mempersiapkan rancang bangun yang terinci untuk masing-masing komponen dari sistem informasi

yang meliputi data dan informasi, simpanan data, metode - metode, prosedur - prosedur, orang - orang, perangkat keras, perangkat lunak dan pengendalian intern.

2.4.3. Tahap-tahap Perancangan Sistem

1. Perancangan Sistem

   Terdiri dari investigasi awal dan studi kelayakan.

2. Kebutuhan sistem

   Terdiri dari operasi analisis sistem dan kebutuhan pemakai, pendekatan dukungan secara teknis, desain konsep dan uji ulang paket, penilaian alternative dan perencanaan.

3. Pengembangan Sistem

   Terdiri dari rancang bangun sistem secara teknis, rancang bangun aplikasi, dan pengesetan prosedur pemakai dan pengendalian latihan untuk pemakai, perancangan, implementasi, perencanaan, konversi, pengetesan sistem.

2.4.4. Alat Bantu Dalam Perancangan Sistem

2.4.4.1 Context Diagram (Diagram Konteks)

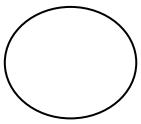
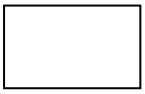
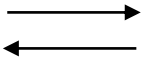
Context Diagram adalah kasus khusus untuk DFD atau bagian dari DFD yang berfungsi untuk memetakan mode lingkungan yang direpresentasikan dengan lingkungan tunggal yang mewakili seluruh sistem. [7]

Context diagram menggarisbawahi sejumlah karakteristik penting dari suatu sistem :

1. Kelompok pemakai, organisasi atau sistem lain dimana sistem komputer melalui komunikasi yang disebut sebagai terminator.
2. Data masuk, data yang diterima sistem dari lingkungan dan harus diproses dengan cara tertentu.

3. Data keluar, data yang dihasilkan dari sistem dan diberikan pada dunia luar.
4. Penyimpanan data, yang digunakan secara bersama antara sistem dengan terminator, data ini dapat dibuat oleh sistem dan digunakan oleh lingkungan dan sistem.
5. Batasan antara sistem dengan lingkungan.

Simbol-simbol yang digunakan :

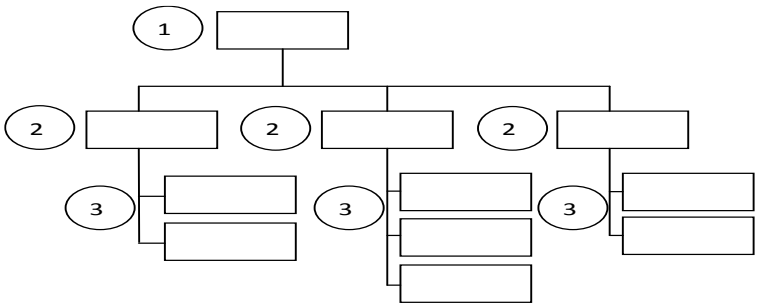
Simbol	Nama / Uraian Simbol
	Lingkaran merupakan symbol untuk keseluruhan sistem
	Segi empat merupakan symbol untuk suatu entity (Entitas)
	Tanda panah merupakan symbol untuk arus data

Tabel 2.2 : Simbol pada Context Diagram

Sumber : [7]

2.4.4.2 Diagram Dekomposisi (*Decompotition Diagram*)

Dekompotition diagram merupakan diagram yang menunjukkan dekomposisi atau struktur fungsional *top-down* suatu sistem. Selain itu dekomposisi diagram juga menyediakan garis besar penggambaran diagram alir yang telah dibuat.



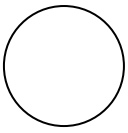
Gambar 2.1 : Contoh Dekomposisi Diagram

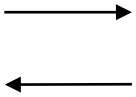
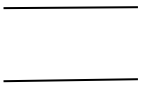
Sumber : [9]

2.4.4.3 Data Flow Diagram Levelled (DFD)

Digunakan untuk menggambarkan sistem sebagai jaringan kerja atau fungsi yang berhubungan satu sama lain dengan aliran-aliran penyimpanan data [7].

Adapun simbol-simbol yang digunakan :

Simbol	Nama / Uraian Simbol
	PROSES Simbol ini digunakan untuk menunjukkan adanya proses transformasi. Proses-proses tersebut selalu menunjukkan suatu perubahan di dalam atau perubahan data. Jadi, aliran data yang meninggalkan suatu proses selalu diberi label yang berbeda dari aliran data yang masuk. Sebuah nama yang jelas memudahkan untuk memahami proses apa yang sedang dilakukan. Digunakan untuk menunjukan transformasi dari masukan menjadi keluaran dalam hal ini sejumlah masukan dapat menjadi hanya satu keluaran atau sebaliknya.

	ARUS DATA (*Data Flow*) Simbol ini menunjukkan perpindahan data dari satu titik ke titik yang lain, dengan kepala tanda panah mengarah ke tujuan data. Karena sebuah tanda panah menunjukkan seseorang, tempat / sesuatu, maka harus digambarkan dalam kata benda. Digunakan untuk menggambarkan gerakan paket data atau informasi dari satu bagian ke bagian lain pada sistem dimana penyimpanan data memiliki lokasi penyimpanan data.
	PENYIMPANAN Penyimpanan data menandakan penyimpanan manual, seperti lemari file/basis data terkomputerisasi. Karena penyimpanan data mewakili seseorang, tempat atau sesuatu, maka diberi nama dengan sebuah kata benda. Penyimpanan data sementara seperti kertas catatan/sebuah file komputer sementara tidak dimasukkan ke dalam diagram aliran data.
	TERMINATOR Melambangkan orang atau kelompok orang.

Tabel 2.3 : Simbol pada DFD

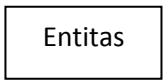
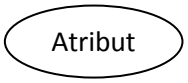
Sumber : [7]

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

ERD adalah model konseptual yang mendeskripsikan hubungan antara penyimpanan dalam DFD. ERD digunakan untuk memodelkan struktur data dan hubungan antar data.

Dengan ERD, model dapat diuji dengan mengabaikan proses yang dilakukan.

Notasi yang digunakan dalam ERD sebagai berikut :

Notasi	Keterangan
	ENTITAS Suatu objek yang dapat diidentifikasi dalam lingkungan pemakai.
	RELASI Menunjukan adanya hubungan diantara sejumlah entitas yang berbeda.
	ATRIBUT Berfungsi mendeskripsikan karakter entitas (atribut yg berfungsi sebagai *key* diberi garis bawah)
	GARIS Sebagai penghubung antara relasi dengan entitas, relasi dan entitas dengan atribut.

Tabel 2.4 : Simbol pada ERD

Sumber : [7]

Diagram ER dibuat secara bertahap, ada dua kelompok pentahapan yang biasa ditempuh didalam pembuatan diagram ER, yaitu :

1. Tahap pembuatan diagram ER awal (*preliminary design*)
2. Tahap optimasi diagram ER (*final design*)

Tujuan dari tahap pertama adalah untuk mendapatkan sebuah rancangan basis data minimal yang dapat mengakomodasi kebutuhan penyimpanan data

terhadap sistem yang sedang ditinjau. Tahap awal ini umumnya mengabaikan anomali-anomali (proses pada basis data yang memberikan efek sampaing yang tidak diharapkan) yang menang ada sebagai suatu fakta. Anomali tersebut biasanya baru dipertimbangkan pada tahap kedua.

Tahap kedua mempertimbangkan anomali-anomali dan juga memperhatikan aspek-aspek efisiensi, performasi dan fleksibilitas. Tiga hal tersebut seringkali dapat saling bertolak belakang. Karena itu, tahap kedua ini ditempuh dengan melakukan koreksi terhadap tahap pertama. Bentuk koreksi yang terjadi dapat berupa pendekomposisian himpunan entitas, penggabungan himpunan entitas, pengubahan derajad relasi, penambahan relasi baru atau perubahan atribut untuk masing-masing entitas dan relasi.

Langkah-langkah teknis yang dapat dilakukan untuk mendapatkan ERD awal adalah :

1. Mengidentifikasi dan menetapkan seluruh himpunan entitas yang akan terlibat.
2. Menentukan atribut-atribut *key* (kunci) dari masing-masing himpunan entitas.
3. Mengidentifikasi dan menetapkan seluruh himpunan relasi diantara himpunan entitas-himpunan entitas yang ada beserta *foreign-key*-nya (kunci asing/kunci tamu).
4. Menentukan kardinalitas relasi untuk setiap himpunan relasi.
5. Melengkapi himpunan entitas dan himpunan relasi dengan atribut deskriptif (atribut yang bukan kunci).

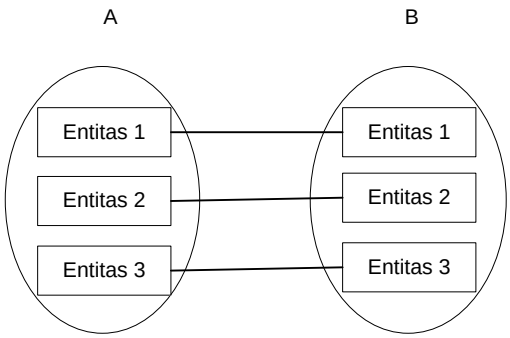
2.4.4.5 Kardinalitas atau Derajat Relasi

Dalam ERD hubungan (relasi) dapat terdiri dari sejumlah entitas yang disebut dengan derajad relasi.

Derajad relasi maksimum disebut dengan kardinalitas sedangkan derajad relasi minimum disebut dengan modalitas. Jadi kardinalitas relasi menunjukkan jumlah maksimum entitas yang dapat berelasi dengan entitas pada himpunan entitas lain. Kardinalitas relasi yang terjadi diantara dua himpunan entitas (misalnya A dan B) dapat berupa [11] :

a. Satu ke satu (*one to one* / 1-1)

Setiap entitas pada himpunan entitas A dapat berelasi dengan paling banyak satu entitas pada himpunan entitas B, demikian juga sebaliknya, dimana setiap entitas pada himpunan entitas B berhubungan dengan paling banyak dengan satu entitas pada himpunan entitas A.

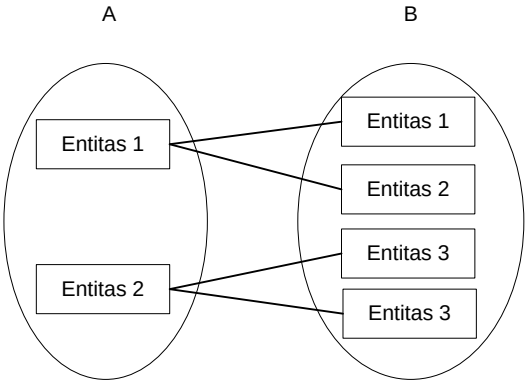


Gambar 2.2 : Relasi satu ke satu

Sumber: [2]

b. Satu ke banyak (*one to many* / 1 – N)

Setiap entitas pada himpunan entitas A dapat berelasi dengan banyak entitas pada himpunan entitas B, tetapi tidak sebaliknya, dimana setiap entitas pada himpunan entitas B berhubungan dengan paling banyak dengan satu entitas pada himpunan A.

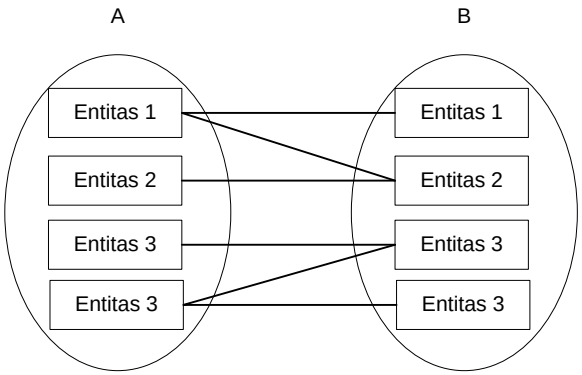


Gambar 2.3 : Relasi satu ke banyak

Sumber: [2]

c. Banyak ke banyak (*many to many* / N – N)

Setiap entitas pada himpunan entitas A dapat berelasi dengan banyak entitas pada himpunan entitas B, demikian juga sebaliknya, dimana setiap entitas pada himpunan entitas B dapat berhubungan dengan banyak entitas pada himpunan entitas A.



Gambar 2.4 : Relasi banyak ke banyak

Sumber: [2]

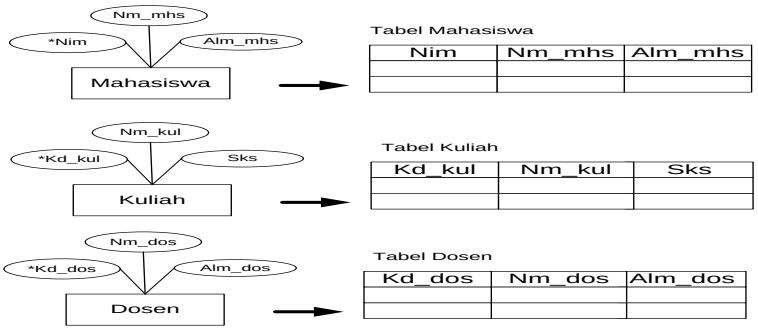
2.4.4.6 *Transformasi* ERD ke Data Fisik atau Tabel

a. Transformasi Umum atau Dasar

Aturan umum dalam pemetaan Mode Data (Level Konseptual dalam Abstraksi Data) yang digambarkan

dengan Diagram E-R menjadi basis data fisik (level fisik dalam abstraksi data) adalah:

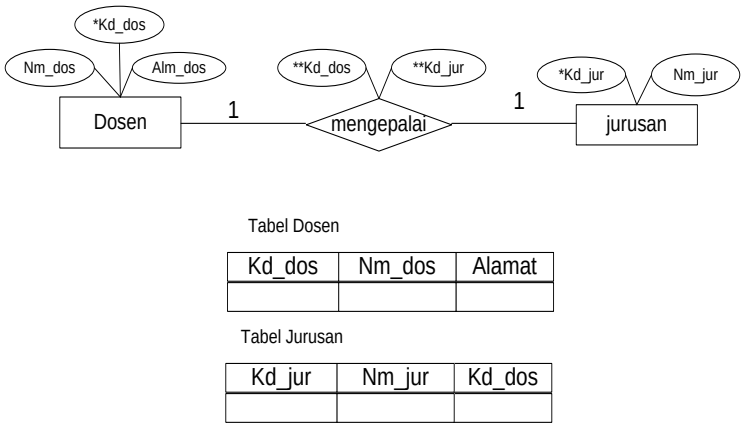
1. Setiap himpunan entitas akan diimplementasikan sebagai sebuah table (file data)



Gambar 2.5 Transformasi Entitas ke Tabel

Sumber: [2]

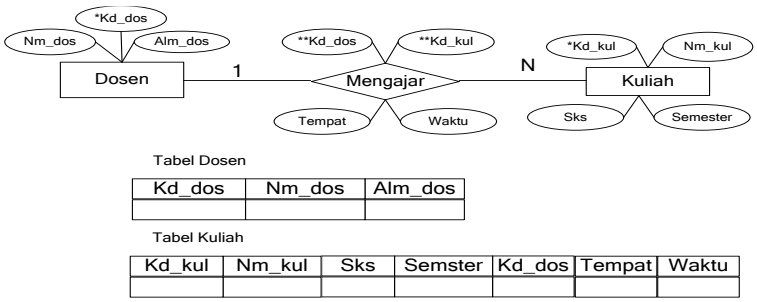
2. Relasi dengan derajat relasi 1-1 (*one to one*) yang menghubungkan dua buah himpunan entitas akan direpresentasikan dalam bentuk penambahan atau penyertaan atribut-atribut relasi ke tabel yang mewakili salah satu dari kedua himpunan entitas.



Gambar 2.6 Transformasi Relasi One to One Tabel

Sumber: [2]

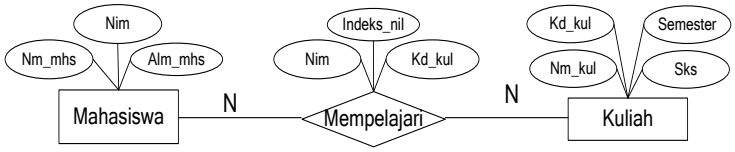
3. Relasi dengan derajat relasi 1-N (*one to many*) yang menghubungkan dua buah himpunan entitas juga akan direpresentasikan dalam bentuk pemberian atau pencantuman atribut *key* dari himpunan entitas pertama yang berderajat 1 ke tabel yang mewakili himpunan entitas kedua yang berderajat M. Atribut *key* dari himpunan entitas pertama ini menjai atribut tambahan bagi himpunan entitas kedua.



Gambar 2.7 Transformasi Relasi One to Many Tabel

Sumber: [2]

4. Relasi dengan derajat Relasi N-N (*many to many*) yang menghubungkan dua buah himpunan entitas, akan diwujudkan dalam bentuk tabel khusus yang memiliki field yang berasal dari *key-key* dari himpunan entitas yang dihubungkan.



Tabel Mahasiswa

Nim	Nm_mhs	Alm_mhs

Tabel Mempelajari

Nim	Kd_kul	Indeks_nil

Tabel Kuliah

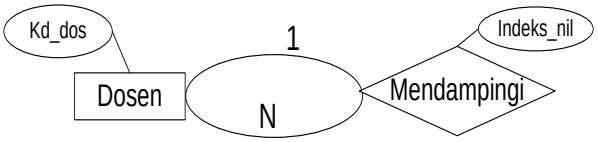
Kd_kul	Nm_kul	Sks	Semester

Gambar 2.8 Transformasi Relasi Many to Many Tabel

Sumber: [2]

b. Implementasi Relasi Tunggal

Implementasi Relasi Tunggal dari atau ke himpunan entitas yang sama dalam Diagram E-R tergantung pada Derajat Relasinya. Untuk Relasi Tunggal dengan Derajat Relasi one to many dapat diimplementasikan melalui penggunaan field key dua kali tapi untuk fungsi yang berbeda.

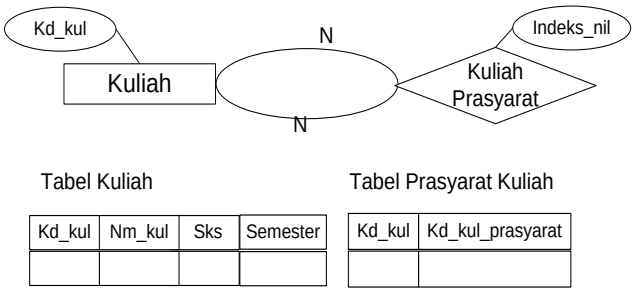


Tabel Dosen

Kd_dos	Nm_dos	Alm_dos	Kd_dos_pend

Sedang relasi yang derajatnya many to many akan diimplementasikan melalui pembentukan tabel baru yang mempresentasikan relasi tersebut. Table baru ini mendapatkan field dari semua atribut relasi jika ada

yang ditambah dengan atribut key dari himpunan entitasnya.

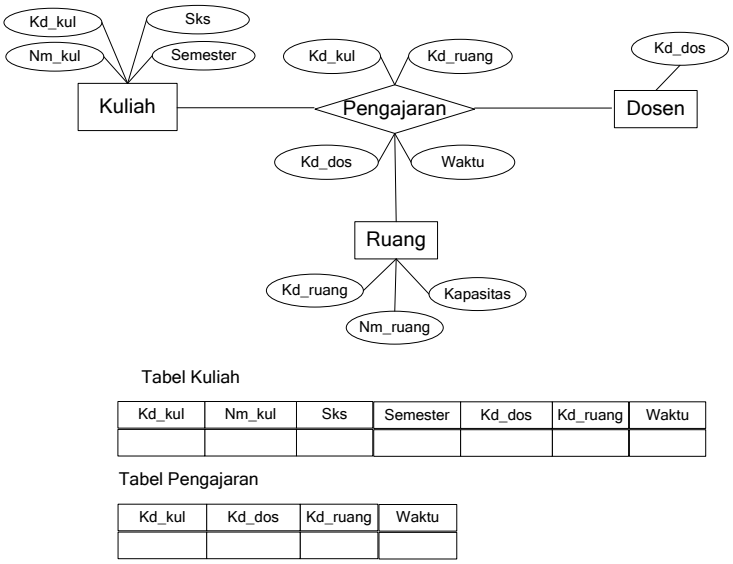


Gambar 2.9 Implementasi Relasi Tunggal

Sumber: [2]

c. Implementasi Relasi Multi Entitas

Secara umum Relasi Multi Entitas yang menghubungkan lebih dari dua himpunan akan diimplementasikan sebagai sebuah table khusus.



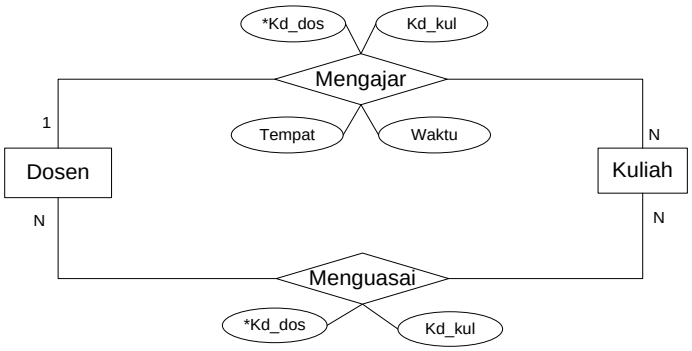
Gambar 2.10 Implementasi Relasi Multi Entitas

Sumber: [2]

d. Implementasi Relasi Ganda

Tidak ada yang istimewa dalam mengimplementasikan relaasi ganda diantara dua himpunan entitas.

Implementasinya kita tinjau pada masing-masing relasi tanpa terikat satu sama lain berdasarkan Derajat Relasi dimasing-masing relasi tersebut.



Karena Derajat Relasi Mengajar adalah many to many, maka field_kd_dos yang berasal dari himpunan entitas Dosen ditambahkan ke tabel kuliah. Sementara untuk relasi menguasai, karena derajat relasinya adalah many to many, maka relasi ini akan dinyatakan dalam tabel khusus dengan dua buah field : Kd_dos dan Kd_kul.

Tabel Dosen

Kd_dos	Nm_dos	Alm_dos

Tabel Menguasai

Kd_dos	Kd_kul

Tabel Kuliah

Kd_kul	Nm_kul	Sks	Semester	Kd_dos

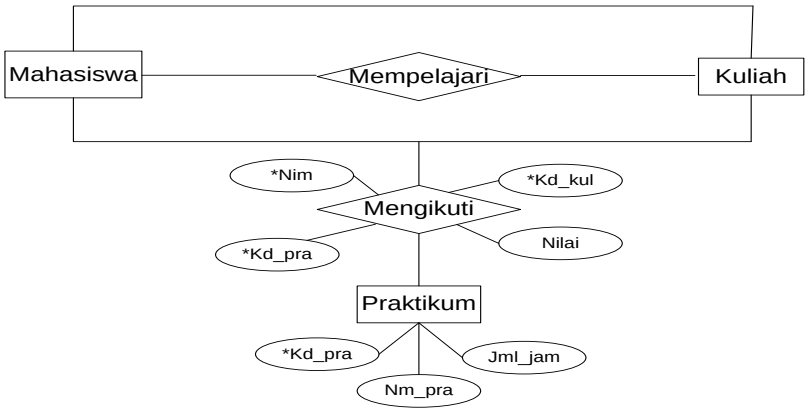
Gambar 2.11 Implementasi Relasi Ganda

Sumber: [2]

e. Implementasi Agregasi

Sesungguhnya Agregasi dapat dipandang sebagaiman relasi pada umumnya yang menghubungkan 2 himpunan. Karena relasi ini dibentuk dari relasi lain

(relasi prasyarat) yang secara kronologis lebih dulu terbentuk, maka pengimplementasiannya juga harus dilakukan setelah relasi prasyarat tersebut terimplementasikan. Selanjutnya kita tinggal kita meninjau Derajat Relasi dari relasi agregasinya.



Relasi mengikuti merupakan bentuk agregasi relasi mempelajari dan entitas praktikum. Karena kesemua derajat relasi yang ada pada Diagram E-R di atas adalah banyak ke banyak, maka baik relasi mempelajari maupun relasi mengikuti masing-masing akan direpresentasikan dalam table khusus atau terpisah.

Tabel Mempelajari

Nim	Kd_kul

Tabel Praktikum

Kd_pra	Nm_pra	Jml_jam

Tabel Mengikuti

Nim	Kd_kul	Kd_pra	Nilai

Gambar 2.12 Implementasi Relasi Agregasi

Sumber: [2]

2.4.4.7 Normalisasi

Normalisasi adalah suatu proses mengubah sebuah tabel yang besar dan kompleks menjadi beberapa buah tabel-tabel yang lebih kecil dan sederhana. [2]

Ada beberapa hal yang perlu diperhatikan dalam normalisasi suatu data yaitu :

1. Field atau Atribut Kunci

   Field kunci adalah sebuah kolom khusus yang memiliki fungsi sebagai pembeda record satu dengan record lainnya.

2. Macam - macam kunci :

   a. Candidat Key atau (Kunci Calon)

      Adalah satu atribut atau field yang mengidentifikasikan secara unik dari suatu kejadian yang sifatnya khusus dari suatu entity.

   b. Primary Key (Kunci Primer)

      Adalah suatu kolom (field) yang menjadi titik acuan pada sebuah tabel, bersifat unik, dalam artian tidak ada satu nilai pun yang sama atau kembar dalam tabel tersebut.

   c. Alternate Key (Kunci Alternatif)

      Adalah kunci kandidat yang tidak dipakai sebagai kunci primer.

   d. Foreign Key (Kunci Tamu)

      Adalah suatu kolom dalam tabel yang digunakan sebagai “kaitan” bagi tabel lainnya sehingga dapat dibuat sebuah hubungan antara tabel tersebut dengan tabel lainnya.

Ada beberapa tahap dalam menormalisasi tabel. Tahapan – tahapan ini diperlukan untuk menyesuaikan table dengan data asli yang ada. Tahapan itu adalah :

1. Bentuk Tidak Normal

   Bentuk ini merupakan kumpulan data yang akan direkam, tidak ada keharusan mengikuti format tertentu, dapat saja data tidak lengkap atau terduplikasi. Data dikumpulkan apa adanya sesuai dengan kedatangannya.

   Contoh :

   File MAHASISWA (No_mhs, Nm_mhs, Nm_PA, nm_MK1, nm_MK2).

   Bentuk diatas adalah bentuk tidak normal karena dalam file tersebut mahasiswa yang mempunyai nomor mahasiswa, nama mahasiswa dan nama PA mengambil 2 mata kuliah, sehingga terjadi perulangan nama mata kuliah 2 kali.

No_mhs	Nm_mhs	Nm_PA	nm_MK1	nm_MK2
0803353	Tatik	Reka	Pancasila	Pascal
0803358	Nindy	Yoga	Matematika	Statistik

Tabel 2.5 : Tabel Mahasiswa Tidak Normal

2. Bentuk Normal ke-1

   Bentuk Normal Pertama mempunyai ciri yaitu setiap data dibentuk dalam file flat, data dibentuk dalam satu record demi record dan nilai dari field berupa “atomic value”. Tidak ada set atribut yang berulang ulang atau atribut bernilai ganda (multi value). Tiap field hanya satu pengertian, bukan merupakan kumpulan data yang mempunyai arti mendua, hanya satu arti saja dan juga bukanlah pecahan kata kata sehingga artinya lain. Atom adalah zat terkecil yang masih memiliki sifat induknya, bila dipecah lagi, maka ia tidak memiliki sifat

induknya, karena hanya akan terdiri dari inti atom dan electron.

Contoh :

Dari tabel mahasiswa diatas diubah menjadi bentuk normal pertama.

No_mhs	Nama_mhs	Nama_PA	nama_MK1
0803353	Tatik	Reka	Pancasila
0803353	Tatik	Reka	Pascal
0803358	Nindy	Yoga	Matematika
0803358	Nindy	Yoga	Statistik

Tabel 2.6 : Bentuk Normal Pertama

3. Bentuk Normal ke-2

Bentuk Normal kedua mempunyai syarat yaitu bentuk data telah memenuhi kriteria bentuk Normal Pertama. Atribut bukan kunci haruslah bergantung secara fungsi pada kunci utama, sehingga untuk membentuk Normal Kedua haruslah sudah ditentukan kunci-kunci field. Kunci field harus unik dan dapat mewakili atribut lain yang menjadi anggotanya.

Contoh :

Dari file mahasiswa, dapat dilihat bahwa kunci primernya adalah field No_mhs. Field Nama_mhs dan field Nama_PA tergantung pada field No_mhs. Sedangkan field nama_MK1 dan MK_2 tidak tergantung pada field No_mhs sehingga bisa dipecah menjadi file yang lain misalnya file AMBILMK.

Sehingga bentuk normal keduanya adalah :

MAHASISWA

No_mhs	Nama_mhs	Nama_PA
0803353	Tatik	Reka
0803358	Nindy	Yoga

AMBILMK

No_mhs	Nama_MK1
0803353	Pancasila
0803353	Pascal
0803358	Matematika
0803358	Statistik

Tabel 2.7 : Bentuk Normal Kedua

4. Bentuk Normal ke-3

Untuk menjadi bentuk Normal Ketiga maka relasi haruslah dalam bentuk Normal Kedua dan semua atribut bukan primer tidak punya hubungan yang transitif. Artinya setiap atribut bukan kunci harus bergantung hanya pada kunci primer secara menyeluruh. Contoh pada bentuk Normal kedua di atas termasuk juga bentuk Normal Ketiga karena seluruh atribut yang ada di situ bergantung penuh pada kunci primernya.

Contoh :

File MAHASISWA dan file AMBILMK sudah merupakan bentuk normal ketiga karena seluruh file yang bukan kunci sudah tergantung pada field kunci yaitu kunci primernya.

2.4.4.8 Kamus Data

Kamus Data (KD) atau *Data Dictionary* (DD) atau disebut juga dengan istilah sistem data dictionary adalah katalog fakta tentang data dan kebutuhan kebutuhan informasi dari suatu sistem informasi. Dengan menggunakan kamus data, analis sistem dapat mendefinisikan data yang mengalir di sistem dengan lengkap. kamus data dibuat pada tahap analisis sistem dan digunakan baik pada tahap analisis maupun pada tahap perancangan sistem. Pada tahap analisis, kamus data dapat digunakan sebagai alat komunikasi antara analis sistem dengan pemakai sistem tentang data yang mengalir di sistem, yaitu tentang data yang masuk ke sistem dan tentang informasi yang dibutuhkan oleh pemakai sistem. Pada tahap perancangan sistem, kamus data digunakan untuk merancang input, merancang laporan-laporan dan database. kamus data dibuat berdasarkan arus data yang ada di DFD. Arus data di DFD sifatnya adalah global, hanya ditunjukkan nama arus datanya saja. Keterangan lebih lanjut tentang struktur dari suatu arus data di DFD secara lebih terinci dapat dilihat di kamus data [7].

Kamus data harus dapat mencerminkan keterangan yang jelas tentang data yang dicatatnya. Untuk maksud keperluan ini, maka kamus data harus memuat hal-hal berikut ini :

1. Nama arus data.

   Karena kamus data dibuat berdasarkan arus data yang mengalir di DAD, maka nama dari arus data juga harus dicatat di kamus data, sehingga mereka yang membaca DAD dan memerlukan penjelasan lebih lanjut tentang suatu

arus data tertentu di DAD dapat langsung mencarinya dengan mudah di kamus data.

2. Alias

   Alias atau nama lain dari data dapat dituliskan bila nama lain ini ada. Alias perlu ditulis karena data yang sama mempunyai nama yang berbeda untuk orang atau departemen satu dengan yang lainnya.

3. Bentuk data.

   Telah diketahui bahwa arus data dapat mengalir :

   a) Dari kesatuan luar ke suatu proses, data yang mengalir ini biasanya tercatat di suatu dokumen atau formulir.
   b) Hasil dari suatu proses ke kesatuan luar, data yang mengalir ini biasanya terdapat di media laporan atau query tampilan layar atau dokumen hasil cetakan komputer.
   c) Hasil suatu proses ke proses yang lain, data yang mengalir ini biasanya dalam bentuk variabel atau parameter yang dibutuhkan oleh proses penerimanya.
   d) Hasil suatu proses yang direkamkan ke simpanan data, data yang mengalir ini biasanya berbentuk suatu variabel.
   e) Dari simpanan data dibaca oleh suatu proses, data yang mengalir ini biasanya berupa suatu *field* (item data).

   Dengan demikian bentuk dari data yang mengalir dapat berupa :

   a) Dokumen dasar atau formulir
   b) Dokumen hasil cetakan computer
   c) Laporan terecetak
   d) Tampilan di layar monitor
   e) Variabel
   f) Parameter

g) *Field*

Bentuk dari data ini perlu dicatat di kamus data, karena dapat digunakan untuk mengelompokkan kamus data ke dalam kegunaannya sewaktu perancangan sistem.

4. Arus data.

   Arus data menunjukkan dari mana data mengalir dan ke mana data akan menuju. Keterangan arus data ini perlu dicatat di kamus data supaya memudahkan mencari arus data ini di DFD.

5. Penjelasan.

   Untuk lebih memperjelas lagi tentang makna dari arus data yang dicatat di kamus data, maka bagian penjelasan dapat diisi dengan keterangan-keterangan tentang arus data tersebut.

6. Periode.

   Periode ini menunjukkan kapan terjadinya arus data ini. Periode perlu dicatat di kamus data karena dapat digunakan untuk mengidentifikasikan kapan input data harus dimasukkan ke sistem, kapan proses dari program harus dilakukan dan kapan laporan-laporan harus dihasilkan.

7. Volume

   Volume yang perlu dicatat di kamus data adalah tentang volume rata-rata dan volume puncak dari arus data. Volume rata-rata menunjukkan banyaknya rata-rata arus data yang mengalir dalam satu periode tertentu dan volume puncak menunjukkan volume yang terbanyak. Volume ini digunakan untuk mengidentifikasikan besarnya simpanan luar yang akan digunakan, kapasitas dan jumlah dari alat input, alat pemroses dan alat output.

8. Struktur data.

   Struktur data menunjukkan arus data yang dicatat di kamus data, terdiri dari item data apa saja.

Fungsi kamus data adalah sebagai berikut :

1. Menjelaskan arti aliran data
2. Mendeskripsikan komposisi paket data
3. Mendefinisikan nilai dan satuan yang relevan
4. Mendeskripsikan hubungan secara detail dalam penyimpanan data yang akan digunakan dalam ERD

Biasanya untuk menunjukkan informasi tambahan di kamus data dipergunakan notasi sebagai berikut ini :

Simbol	Keterangan
=	Terbentuk dari, terdiri dari, artinya, atau sama dengan.
+	Dan.
[ ]	Salah satu dari
|	Sama dengan simbol [ ] .
N{ }M	Iterasi (elemen data dalam kurung beriterasi mulai minimum N kali dan maksimum M kali.
( )	Optional (boleh ada atau tidak)
*	Keterangan setelah tanda ini adalah komentar

Tabel 2.8 : Simbol pada Kamus Data

[Sumber : 7]

2.5. Konsep Dasar MySQL

2.5.1. Pengertian MySQL

MySQL adalah sebuah system manajemen database relasi (relation database management system) yang bersifat "terbuka" (open source). Terbuka maksudnya adalah MySQL boleh di-download oleh siapa saja, baik versi kode program aslinya (source kode program) maupun versi binernya (executable program) dan bias digunakan secara (relatif) gratis baik untuk dimodifikasi sesuai dengan kebutuhan seseorang maupun sebagai suatu program aplikasi computer. [4]

MySQL adalah sebuah program database server yang mampu menerima dan mengirimkan datanya dengan sangat cepat, multi user serta menggunakan perintah standar SQL (Structured Query Language). [9]

SQL (*Structured Query Lenguage*) dibagi menjadi dua bentuk *Query*, yaitu: [13]

1. DDL *(Data Definition Language)*

   DDL adalah sebuah Metode Query SQL yang berguna untuk mendefinisikan data pada sebuah database, adapun Query yang dimiliki adalah :

   - *CREATE* : Digunakan untuk melakukan pembuatan database dan tabel
   - *DROP* : Digunakan untuk melakukan penghapusan tabel maupun database.
   - *ALTER* : Digunakan untuk melakukan pengubahan struktur tabel yang telah dibuat, baik menambah Field (*add*), mengganti nama Field (*change*) atau menamakannya kembali (*rename*), serta penghapusan (*drop*).

2. DML *(Data Manipulation Language)*

DML adalah sebuah Metode *Query* yang dapat digunakan apabila DDL telah terjadi, sehingga fungsi dari *Query* ini adalah untuk melakukan pemanipulasian database yang telah ada atau telah dibuat sebelumnya. Adapun *Query* yang termasuk didalamnya adalah :

- *INSERT* : Digunakan untuk melakukan penginputan atau pemasukan data pada tabel database
- *UPDATE* : Digunakan untuk melakukan pengubahan terhadap data yang ada pada tabel.
- *DELETE* : Digunakan untuk melakukan penghapusan data pada tabel. Penghapusan ini dapat dilakukan secara sekaligus maupun hanya beberapa *Recordset*.
- *SELECT* : Digunakan untuk menampilkan isi data pada tabel

2.5.2. Implementasi Basis Data

a. Membuat Database

Untuk membuat database baru, maka perintahnya adalah :

Mysql> Create Database namadatabase;

b. Mengetahui Nama Basis Data yang Sedang Aktif

Untuk mengetahui nama basis data yang sedang kita masuki (gunakan), tuliskan perintah database().

Mysql> select database ();

c. Manampilkan nama-nama Basisdata

Database yang telah ada di dalam sistem dapat ditampilkan dengan menggunakan perintah:

Mysql> show databases ();

d. Memilih Database yang akan digunakan

Untuk menggunakan atau memilih suatu database digunakan perintah:

Mysql> Use namabasisdata;

e. Melihat nama-nama tabel dalam suatu basisdata

Untuk melihat tabel yang ada di dalam sebuah database digunakan perintah:

Mysql> show tables;

f. Membuat Tabel Baru

Untuk dapat membuat suatu tabel. Dalam hal ini yaitu dengan menggunakan pernyataan CREATE TABLE:

Contoh:

Akan dibuat tabel mhs yang berisi data nim, nama dan tanggal lahir:

MySql> create table mhs (nim char(4) not null primary key,

->nama char (20),tgl_lahir date);

Query OK, 0 rows affected (0.01 sec)

g. Menampilkan Struktur Suatu Tabel

Struktur dari sebuah tabel dapat ditampilkan dengan menggunakan perintah DESCRIBE, bentuk perintahnya adalah:

Mysql> Describe namatabel;

Contoh

mysql> desc mhs;

Field	*Type*	*Null*	*Key*	*Default*	*Extra*
nim	*char (8)*		*PRI*		
nama	*char (20)*	*YES*		*NULL*	

alamat	***char (25)***	***YES***		***NULL***	
kota	***char (20)***	***YES***		***NULL***	
sex	***char (1)***	***YES***		***NULL***	
agama	***cahar (10)***	***YES***		***NULL***	
tgl_lahir	***date***	***YES***		***NULL***	

3 rows in set (0.02 sec)

Tabel 2.9 : Menampilkan Struktur suatu Tabel

h. Mengisi Data

Data yang telah terbentuk dapat diisikan data dengan perintah :

Mysql> insert into namatabel value();

Contoh:

```
mysql> insert into mhs values
    ->('1111','Nindy','jl.melati','Semarang','W','1',
    '1984- 08-08'),
    ->('2222','Tian','jl.bromo',Semarang','W','2',
    '1978-19-12'),
    ->('3333','Tatik','jl.aceh','Semarang','W','2',
    '1987-17-12');
Query OK, 3 rows affected (0.00 sec)
Records : 3 Duplicates : 0 Warnings : 0
```

i. Menampilkan Data

Tabel yang telah dibuat dan diisikan dengan data dapat ditampilkan isinya (informasinya) dengan menggunakan perintah SELECT.

Contoh:

mysql> select * from mhs;

Nim	*Nama*	*Alamat*	*Kota*	*Sex*	*Agama*	*Tgl_lahir*
1111	*Nlndy*	*Jl.melati*	*Semarang*	*W*	*1*	*1984-08-08*
2222	*Tian*	*Jl.bromo*	*Semarang*	*L*	*2*	*1978-19-12*
3333	*Tatik*	*Jl.aceh*	*Semarang*	*W*	*2*	*1985-19-02*

4 rows in set (0.00 sec)

Tabel 2.10 : Menampilkan Data

j. Menghapus Data

Untuk menghapus record, maka gunakan pernyataan DELETE dengan syntax sebagai berikut:

Mysql> DELETE FROM nama_tabel WHERE kondisi;

Contoh:

mysql> Delete from mhs where nim = '1111';

Hasilnya:

mysql> select * from mhs;

Nim	*Nama*	*Alamat*	*Kota*	*Sex*	*Agama*	*Tgl_lahir*
2222	*Tian*	*Jl.bromo*	*semarang*	*L*	*2*	*1978-19-12*
3333	*Tatik*	*Jl.aceh*	*semarang*	*W*	*2*	*1985-02-19*

rows in set (0.00 sec)

Tabel 2.11 : Hasil Penghapusan Data

k. Mengganti Tabel

1. Mengganti Nama suatu Field

Apabila kita mengganti nama suatu field, maka dapat digunakan perintah rename dengan format:

Alter Table <namatable> Rename <namafield_baru>

Contoh:

Akan mengganti nama tabel mhs menjadi tabel pegawai:

mysql> Alter Table mhs Rename pegawai;

2. **Mengganti nama field serta ukuran**

   Untuk mengganti nama serta ukuran suatu field, gunakanlah perintah change dengan format seperti berikut:

   Alter Table <namatable> Change <namafield_lama> <namafield_baru>tipe_data_baru;

   Contoh:

   Mengganti nama field NIM menjadi NOINDUK yang panjangnya 14 menjadi 15:

   mysql> Alter Table mhs change nim noinduk char(15);

3. **Mengganti tipe data (ukuran) field**

   Untuk mengganti tipe data (ukuran) field saja, kita dapat menggunakan perintah modify dengan format perintahnya adalah:

   Alter Table <namatable> modify <namafield>
   <tipe_data_baru>;

   Contoh:

   Akan diganti tipe data dari field sal, yang semula int menjadi decimal (8,2):

   mysql> Alter Table emp modify sal decimal(8,2);

l. Mengubah Record

Untuk mengubah record, maka gunakan pernyataan UPDATE.

Contoh:

Nip	Gaji
11111	2000000
22222	3000000
33333	4000000
44444	5000000
55555	6000000

Sekarang cobalah untuk memberikan perintah seperti berikut:

UPDATE gajipeg

SET gaji = 5000000

Hasilnya:

Nip	Gaji
11111	5000000
22222	5000000
33333	5000000
44444	5000000
55555	5000000

Table 2.12 : Mengubah Record

2.5.3. Struktur Data

Struktur dasar dari ekspresi SQL terdiri dari tiga klausa yaitu : *select, from* dan *where* . [8]

Klausa *Select* digunakan untuk menetapkan daftar atribut (*field*) yang diinginkan sebagai hasil *query*.

1. Klausa *From* digunakan untuk menetapkan tabel atau gabungan tabel yang akan ditelusuri selama *query* data dilakukan.

2. Klausa *Where* yang sifatnya opsional, digunakan sebagai predikat (kriteria) yang harus dipenuhi dalam memproleh *query*.

Cara penulisan dari ekspresi SQL dasar dengan klausa tersebut adalah:

Select A1 [, A2,…..An] From t1 [, t2,….tn] [Where P];

Dimana :

- A1, A2, ..., An merupakan daftar atribut.
- t1, t2, ..., tn merupakan daftar tabel.
- P merupakan predikat *query*.
- [ ] merupakan tanda opsional yang boleh digunakan, boleh tidak.

1. Klausa *Select*

Jika ingin menampilkan nama mahasiswa yang ada di tabel mahasiswa, maka dapat menggunakan perintah SQL berikut ini :

select nama_siswa from mahasiswa;

Akan menampilkan semua data nama mahasiswa yang ada. Jika katakanlah di dalam tabel terdapat dua mahasiswa dengan nama yang sama tapi NIMnya berbeda, maka nama tersebut juga akan tampil dua kali. Jika kita mengharapkan agar nilai atribut yang tampil bersifat unik perintahnya :

select distinct nama_mhs from mahasiswa;

Perlu diperhatikan keunikan diatas hanya untuk nilai atribut yang disebutkan dalam klausa *select*, bukan pada keseluruhan atribut yang ada di dalam tabel yang disebutkan dalam klausa *from*.

Pada tampilan hasil *query* yang disusun berbentuk tabular, atribut yang disebutkan pada klausa *select* akan dijadikan sebagai *header* (kepala tampilan tabular). Untuk dapat

mengganti tampilan *header* tanpa mengganggu proses dan hasil *query*-nya dengan menambahkan klausa *as* sebagai berikut :

Select nim, nama_mhs as nama from mahasiswa;

2. Klausa *Where*

Contoh query : untuk menampilkan semua atribut untuk mahasiswa dengan NIM=’980002’.

Query-nya ditulis sebagai berikut :

Select * From mahasiswa Where nim = ‘980002’;

Penggunaan tanda kutip digunakan terhadap nilai yang bertipe string (yang harus disesuaikan dengan tipe dari atribut NIM)

Misal ingin menampilkan semua mata pelajaran yang diselenggarakan di semester 1 tetapi yang jumlah jam-nya lebih besar dari 2, *query* nya ditulis sebagai berikut :

select * from kuliah where semester = 2 and sks > 2;

Misal untuk menampilkan *record* belajar yang diselenggarakan antara semester 2 hingga semester 4 :

select * from kuliah where semester between 2 and 4;

Khusus untuk atribut yang bertipe *string* dapat melakukan pencarian dengan pola tertentu, dengan memanfaatkan karakter ‘%’ yang berarti cocok untuk semua *substring* atau ‘_’ yang berarti cocok untuk semua karakter pada posisi yang sesuai dan tambah klausa like pada klausa where.

Berikut ini adalah perintah untuk menampilkan *record-record* mahasiswa yang namanya diawali dengan huruf ‘A’

Select * From mahasiswa Where nama_mhs like ‘A%’;

3. Klausa *From*

Contoh *query* : "Untuk menampilkan data kuliah beserta dosen-dosen yang mengajarkannya".

Dalam SQL ditulis :

Select * From kuliah, dosen Where kuliah.kode_dos= dosen.kode_dos;

4. Pengurutan Hasil *Query*

Contoh *query* : " Untuk menampilkan *record-record* mahasiswa berdasarkan urutan namanya"

Select * From mahasiswa Order by nama_mhs;

Untuk menampilkan *record-record* mahasiswa berdasarkan urutan namanya tapi secara menurun (dari mahasiswa termuda hingga mahasiswa tertua) maka dalam SQL ditulis:

Select * From mahasiswa Order by nama_mhs desc;

2.5.4. Manipulasi Data

Meliputi penambahan *record* baru, pengubahan nilai atribut, dan penghapusan *record* di dalam suatu tabel [8].

1. Penambahan Record

Sintaks SQL untuk penambahan record baru ke sebuah tabel adalah

insert into *t [(A1, A2, ..., An)* values *(v1, v2, ..., vn);*

dimana :

- t adalah nama tabel yang akan mengalami penambahan *record*
- A1, A2, ..., An adalah nama-nama atribut yang akan diisi nilai
- v1,v2, ..., vn adalah nilai-nilai yang akan mengisi atribut-atribut tersebut.

Contoh untuk penambahan *record* baru ke tabel mahasiswa :

insert into mahasiswa (nim, nama_mhs, alamat_mhs) **Values** (‘0304715’,’Tian Indriatmanto’, ‘Jl. Gajah Mada No 19’);

2. Pengubahan Record

Sintaks SQL untuk pengubahan nilai atribut dari sebuah tabel adalah :

update *t* set *assignment* [where P];

dimana :

- t adalah nama tabel yang akan mengalami pengubahan *record.*
- *assignment* adalah ekspresi pemberian nilai baru untuk suatu atribut yang akan kita ubah
- P merupakan predikat atau kriteria untuk pemilihan *record* yang akan dikenai perubahan jika klausa *where* ini tidak digunakan, maka perubahan akan dilakukan pada semua *record* di dalam tabel t.

Contoh untuk mengubah nilai atribut sks untuk mata kuliah tertentu :

update kuliah **set** sks = ‘4’ **where** kode_kul = ‘TA-127’;

3. Penghapusan Record

Sintaks SQL untuk penghapusan record dari sebuah tabel adalah

delete from t where p;

dimana :

- t adalah nama tabel yang akan mengalami penghapusan *record.*

- P merupakan predikat atau kriteria untuk menentukan *record* mana saja yang akan dikenai penghapusan jika klausa *where* ini tidak digunakan, maka penghapusan akan dilakukan pada semua *record* di dalam tabel t.

  Contoh :

  a. Hapus *record* kuliah tertentu, ditulis :

     Delete from kuliah where kode_kul = 'TA-127';

  b. Hapus beberapa *record* di tabel kuliah

     Delete from kuliah where kode_kul like 'MA%;'

  c. Hapus semua *record* dari tabel kuliah

     Delete from kuliah;

2.5.5. Fungsi Agregasi

Yang termasuk fungsi-fungsi agregasi adalah :

1. *count* untuk mendapatkan nilai banyak record hasil *query*

   Contoh untuk menampilkan banyaknya *record* mahasiswa

   Select count (*) **From** Mahasiswa;

2. *sum* : total untuk mendapatkan nilai total suatu atribut numerik hasil *query.*

   Contoh untuk menampilkan total sks untuk kuliah disemester 2

   Select sum (sks) From kuliah **Where** semester = 2;

3. *Average* : avg untuk mendapatkan nilai rata-rata suatu atribut numenrik hasil *query.*

   Contoh untuk menampilkan rata-rata sks untuk semua matakuliah

   Select avg (sks) **From** kuliah;

4. *Max* untuk mendapatkan nilai terbesar dari hasil *query*
   Contoh untuk menampilkan nilai terbesar yang diperoleh mahasiswa untuk mata kuliah dengan kode kuliah ‘TA-644’

   Select max (index_nilai) **From** mahasiswa
   Where kode_kul = ‘TA-644’;

5. *Min* untuk mendapatkan nilai terkecil dari hasil *query*
   Contoh untuk menampilkan tanggal lahir paling tua yang ada didalam tabel mahasiswa

   Select min(tgl_lahir) **From** mahasiswa;

   Fungsi agregasi ini dapat pula dikombinasikan dengan klausa *group by* yang menyatakan adanya pengelompokan *record-record* hasil *query*. Jika kita ingin menampilkan banyak record dan total sks untuk mata kuliah yang dikelompokan berdasarkan nilai semesternya, maka ekspresi SQL ini dapat kita gunakan :

   Select semester, count (*), sum (sks)
   From kuliah
   Group by semester
   Order by semester;

   Penggunaan klausa *order by* diatas hanya untuk tujuan agar hasil *query*-nya diurutkan berdasarkan nilai atribut semesternya.
   Fungsi agregasi harus selalu diikuti dengan tanda kurung dan nama atribut diantara tanda kurung tersebut. Karena fungsi *count* lebih berorientasi pada banyaknya *record* hasil *query*, maka boleh untuk tidak menuliskan nama atribut spesifik di antara kurung setelah fungsi ini.

2.5.6. Bahasa Basis Data (Data Definition Language)

DDL berkaitan dengan perintah-perintah untuk pendefinisian objek-objek basis data. Salah satu objek terpenting adalah tabel. Berikut ini adalah sintaks SQL untuk melakukan pembuatan tabel baru didalam basis data :

create table *t (A1, D1, A2, D2, ..., An, Dn)*;

Dimana :

- t adalah nama tabel yang akan dibuat
- A1, A2, ..., An adalah nama-nama atribut yang akan terdapat di dalam tabel t.
- D1, D2, ..., Dn adalah domain nilai masing-masing atribut tersebut yang ditentukan berdasarkan tipe datanya.

Sebelum perintah tersebut digunakan, terlebih dahulu harus mengetahui tipe data apa saja yang dapat kita gunakan. Dalam SQL-92 tipe data yang menjadi standar adalah :

a. Char(n) untuk atribut yang bernilai string dengan panjang tetap sebesar n karakter (*fixed-length character*).
b. Varchar(n) untuk atribut yang bernilai string dengan panjang fleksibel, tapi maksimal sebanyak n karakter.
c. Int : *integer* untuk atribut yang bernilai *integer* 2 byte
d. Smallint untuk atribut yang bernilai *integer* 1 byte
e. Numeric (p,d) : untuk atribut yang bernilai pecahan *fixed-point* dengan panjang P dijit termasuk tanda dan d untuk bilangan pecahan.
f. Real, double precision : untuk atribut yang bernilai pecahan *floating-point*
g. Float (n) : untuk atribut yang bernilai pecahan *floating-point* dengan presisi n dijit
h. Date untuk atribut yang bernilai penanggalan

i. Time untuk atribut yang bernilai waktu terdiri atas jam, menit dan detik.

Berikut ini adalah contoh perintah SQL untuk membuat tabel mahasiswa.

create table mahasiswa (nim **char** (6), nama_mhs **varchar** (30), alamat_mhs **varchar** (60), tanggal_lahir **date**);

Jika terdapat indeks Primer berdasarkan atribut tertentu di dalam tabel, maka klausa *primary key* dapat digunakan. Berikut contoh ekspresi SQL untuk pembuatan tabel Mahasiswa sekaligus dengan pendefinisian Indeks Primer berdasarkan nim :

create table mahasiswa (nim **char** (6), nama_mhs **varchar** (30), alamat_mhs **varchar** (60), tanggal_lahir **date**, **primary key** (nim));

Struktur sebuah tabel juga dapat diubah tanpa harus menghapus dan kemudian membangunnya kembali dengan definisi struktur yang baru. Perintah pengubahan struktur selain lebih praktis juga tidak mengakibatkan hilangnya data yang sudah ada di dalam tabel. Perubahan struktur ini dapat berupa penambahan atribut atau pengurangan/penghapusan atribut tertentu. Sintaks SQL untuk perubahan struktur tabel yang berbentuk penambahan atribut baru ke tabel t adalah :

alter table *t* add *A D*;

Dimana t mewakili nama tabel, A mewakili nama atribut dan D mewakili tipe data untuk atribut A tersebut.
contoh ekspresi SQL untuk penambahan atribut baru bernama ip di tabel mahasiswa :

alter table mahasiswa **add** ip numeric(5,2);

Untuk penghapusan atribut dari table t, sintaks SQL-nya adalah :

alter table *t* drop *A*;

Jika atribut ip ingin dihapus dari tabel mahasiswa, ekspresi SQL-nya:

alter table *mahasiswa* drop *ip*;

2.5.7. Constraint

Constraint berfungsi memaksakan aturan tertentu pada tabel atau kolom. Kita dapat menggunakan constraint untuk : [16]

- Memaksakan aturan pada level tabel ketika dilakukan operasi insert, update atau delete terhadap tabel tersebut. Constraint harus dipenuhi agar operasi-operasi tersebut dapat dilakukan dengan sukses.
- Mencegah penghapusan terhadap tabel apabila terdapat ketergantungan dari tabel lain.
- Menyediakan aturan-aturan untuk development tools Oracle lainnya seperti Oracle Forms dan Reports.

Jenis-jenis Constraint sebagai berikut :

1. Constraint PRIMARY KEY
   - Constraint PRIMARY KEY menciptakan kunci utama pada tabel. Satu tabel hanya boleh memiliki satu kunci utama.
   - Constraint PRIMARY KEY adalah kolom atau kombinasi kolom yang secara unik membedakan baris data satu dengan lainnya dalam suatu tabel.
   - Constraint ini memastikan bahwa nilai kolom yang merupakan bagian dari kunci utama haruslah unik dan tidak mengandung nilai null.
   - Constraint PRIMARY KEY dapat didefinisikan pada constraint level kolom atau level tabel. Kunci utama yang

terdiri dari beberapa kolom diciptakan pada definisi level tabel.

- UNIQUE index otomatis diciptakan untuk kolom yang memiliki constraint ini.

2. Foreign Key

Kunci tamu didefinisikan pada tabel anak, dan mengandung kolom yang diacunya pada tabel induk. Kunci tamu didefinisikan dengan kombinasi perintah-perintah berikut:

- FOREIGN KEY digunakan untuk mendefinisikan kolom pada tabel anak pada level constraint kolom.
- REFERENCES menentukan tabel dan dan kolom pada tabel induk.
- ON DELETE CASCADE menyatakan bahwa ketika baris data pada table induk dihapus, maka baris data pada table anak yang mengacu kepadanya juga akan dihapus.

3. Constraint UNIQUE

- Constraint UNIQUE menandakan bahwa kolom atau kombinasi kolom bernilai unik. Constraint ini melarang adanya nilai duplikat.
- Nilai null diperbolehkan apabila kunci unik berbasis kolom tunggal.
- Constraint UNIQUE dapat didefinisikan pada constraint level kolom atau level tabel.
- Kunci unik yang terdiri dari beberapa kolom diciptakan pada definisi level tabel.
- UNIQUE index otomatis diciptakan untuk kolom yang memiliki constraint ini.

4. Not Null

   Constraint NOT NULL melarang nilai null untuk suatu kolom. Kolom tanpa constraint NOT NULL secara default dapat mengandung nilai null. Constraint ini hanya dapat didefinisikan pada level kolom.

5. Check

   Constraint CHECK mendefinisikan kondisi yang harus dipenuhi oleh setiap baris data. Kondisi dapat menggunakan bentuk yang sama dengan kondisi-kondisi query dengan perkecualian:

   - Mengacu pada kolom CURRVAL, NEXTVAL, LEVEL, atau ROWNUM.
   - Memanggil fungsi SYSDATE, UID, USER, atau USERENV.
   - Query yang mengacu kepada nilai dari baris data lain.

     Constraint CHECK dapat didefinisikan pada constraint level kolom atau level tabel.

2.5.8. Index

Index adalah struktur data khusus yang dibuat untuk meningkatkan kinerja database.

Indeks dalam database dapat diumpamakan seperti indeks dalam sebuah buku yang tebal, sehingga item tertentu dapat ditemukan dengan cepat. Sebuah indeks dalam basis data berfungsi untuk mempercepat pencarian data berdasarkan kolom tertentu. Misal sebuah perintah : [13]

*SELECT * FROM pegawai* WHERE *nip =’* 19571102001*’;*

Jika nip tidak dijadikan sebagai indeks, pencarian data akan dilakukan terhadap seluruh tabel, sama seperti kalau akan mencari sesuatu dalam buku tetapi buku tersebut tidak dilengkapi dengan

indeks. Namun sekiranya indeks yang berkaitan dengan nip ada, maka sistem akan menemukannya dengan cepat.

a. Menciptakan index

   MySQL> Create Index nip_idx on pegawai (nip);

b. Menciptakan index yang unik

   MySQL> Create unique index pegawai_idx on pegawai (nip, kd_gol);

2.5.9. View

View mirip dengan *Stored Procedure*. Dalam implementasinya, *view* biasa digunakan untuk menyederhanakan *query* yang kompleks untuk keperluan *reporting*. *View* dapat terdiri dari satu atau lebih *query*, termasuk *nested query*. *Record* pada sebuah *view* ada yang dapat dimanipulasi, dan ada pula yang tidak, tergantung DBMS yang digunakan.

Contoh :

Membuat view promosi pegawai :

```
Mysql> CREATE VIEW `skripsi`.`peg_promosi`
       AS
       Select b. kd_promosi, a.nip, a.nama, c.nm_gol,
       b.tmpt_promosi, a.status, a.SK
       From pegawai a , promosi b, golongan c
       Where b.nip = a.nip and a.kd_gol=c.kd_gol and
       b.tmpt_promosi= 'Semarang'
       group by b.kd_promosi
```

Hasilnya :

Kd_promosi	Nip	Nama	Nm_gol
P005	19600525005	Beni Sulasto	Penata Tingkat I
P012	19651222012	Andi Wicaksono	Penata Tingkat I

Tmpt_promosi	Status	SK
Semarang	Kawin	057.5/279/PEG.06
Semarang	Kawin	045.1/250/PEG.02

Tabel 2.13 : Menampilkan View

2.5.10. Stored Procedure

Store procedure adalah bahasa pemrograman yang bisa disimpan pada suatu basis data sehingga memungkinkan pengolahan terhadap basis data tersebut secara langsung untuk operasi-operasi yang lebih kompleks. Keuntungan dari (SP) antara lain : [4]

Store procedure (prosedur tersimpan) adalah suatu modul yang berisi kumpulan pernyataan SQL yang ditujukan untuk melaksanakan tugas tertentu dan letaknya ada pada server. Modul ini bias dipanggil oleh klien, sedangkan pengeksekusian dilakukan di server. [13]

Keuntungan dari Store procedure adalah sebagai berikut :

1. Meningkatkan kinerja karena mengurangi pengiriman kode dari klien ke server, mengingat modul berada pada server
2. Meningkatkan keamanan karena pengaksesan data tertentu ditangani dalam server, tidak melalui pengaksesan secara langsung oleh klien.
3. Meningkatkan integritas data (konsistensi data) saat sejumlah aplikasi memanggil prosedur tersimpan yang sama.

2.5.11. Stored Function

Stored function (fungsi tersimpan) menghasilkan nilai ketika dipanggil dan tentu saja seperti fungsi biasa dipanggil di dalam suatu pernyataan (misalnya dalam select). Nilai yang dihasilkan oleh fungsi tersimpan biasanya disebut nilai balik (return value).

2.5.12. Trigger

Trigger adalah mekanisme kontrol yang merupakan keistimewaan Sql server . Trigger sama seperti sekumpulan perintah Transact-SQL yang secara otomatis dijalankan apabila ada perintah INSERT, DELETE, atau UPDATE yang dijalankan di dalam tabel. Aplikasi utama dari trigger adalah pembuatan metode validasi dan batasan akses ke dalam database.[17]

Trigger merupakan kumpulan perintah SQL yang secara otomatis dijalankan untuk merespon sebuah perintah tertentu. Biasanya, secara fisik *trigger* menjadi satu dengan *table* atau *view*.

Perintah – Perintah yang dapat dilakukan Trigger antara lain adalah :

a. Membuat isi dari kolom yang diambil dari kolom yang lain.
b. Membuat mekanisme validasi yang mencakup query pada banyak tabel.
c. Membuat log untuk mendaftarkan penggunaan tabel.
d. Meng-update tabel - tabel lain apabila ada penambahan atau perubahan lain di dalam tabel yang sedang aktif.

   Contoh perintah membuat Trigger ada 2 cara yaitu :

   a. Menggunakan Query Analizer

      Create trigger [trigger name] on [nama tabel]
      For insert, update, delete
      As perintah

   b. Menggunakan Enterprise Manager

- Menuliskan perintah trigger input

Create trigger input-tsupplier on (tsupplier)
For insert
As
Print 'data telah diinputkan'

- Menuliskan perintah trigger delete

Create trigger delete_tsupplier on (tsupplier)
For delete
As
Print "data telah dihapus'

- Menuliskan perintah trigger update

Create trigger update_tsupplier on (tsupllier)
For update
As
Print 'data telah diperbaharui'

2.6. Konsep Dasar Administrasi Kepegawaian

2.6.1. Pengertian Administrasi

Kata administrasi berasal dari kata *administrate* yang artinya melayani, pengabdian atau membantu, yang berarti melakukan suatu kegiatan tertentu untuk mencapai suatu tujuan tertentu pula yaitu untuk memenuhi hasrat dan kebutuhan manusia itu sendiri yang relatif bersifat jamak. Dalam bahasa inggris berasal dari kata administration yang artinya kegiatan yang dilakukan (oleh para administrator profesional) untuk mengendalikan suatu usaha (negara) agar tujuan yang diinginan tercapai. administrator adalah orang yang mengepalai/memimpin administrasi, misal : seorang kepala sekolah, kantor, departemen, badan dan sebagainya.[65]

Dengan demikian maka administrasi adalah Organisasi dan manajemen dari pada sumber daya yang dilakukan secara teratur guna mencapai tujuan.

2.6.2. Pengertian Kepegawaian

Kepegawaian adalah sifat-sifat atau segala sesuatu yang mengenai pegawai, dimana pegawai adalah orang-orang yang bekerja pada pemerintah atau perusahaan. Dalam sistem kepegawaian, kepegawaian menerima data-data kepegawaian dari pegawai-pegawai, untuk kelancaran pelaksanaan tugas dipandang perlu mengeluarkan pola-pola pembinaan dan pengendalian administrasi kepegawaian.[5]

2.6.2.1 Manajemen Kepegawaian

Menurut Marry Follet, manajemen adalah seni untuk menyelesaikan pekerjaan melalui penagaturan orang lain untuk melaksanakan tugas itu sendiri.

Menurut Edwin b Flippo, manajemen kepegawaian adalah perencanaan, pengorganisasian, pengarahan dan pengawasan daripada pengadaan, pengembangan, kompensasi, integrasi dan pemeliharaan orang-orang untuk menunjang tujuan-tujuan organisasi, individu dan sosial.[5]

2.6.2.2 Definisi Kenaikan Pangkat

Kenaikan pangkat adalah kenaikan kedudukan yang menunjukkan tingkat seseorang Pegawai Negeri Sipil berdasarkan jabatannya dalam rangkaian susunan kepegawaian dan digunakan sebagai dasar penggajian.

Kenaikan pangkat adalah penghargaan yang diberikan kepada PNS atas prestasi kerja dan pengabdiannya kepada Negara. Penentuan pangkat diatur dalam Peraturan Pemerintaha Nomor 7 Tahun 1977 dan Peraturan Pemerintah Nomor 11 Tahun 2003.[14]

2.6.2.3 Definisi Mutasi

Mutasi adalah perubahan dari suatu bentuk dasar menjadi suatu bentuk yang lain, atau perpindahan dari suatu tempat ke tempat lainnya dalam suatu waktu.[5]

2.6.2.4 Definisi Cuti

Yang dimaksud dengan cuti adalah tidak masuk kerja yang diijinkan dalam jangka waktu tertentu. Pegawai Negeri Sipil yang bekerja dalam jangka waktu tertentu perlu istirahat atau cuti. Pemberian cuti dimaksud untuk mengembalikan kesegaran jasmani dan rohani Pegawai Negeri Sipil yang bersangkutan.

Cuti adalah hak Pegawai Negeri Sipil, kecuali cuti diluar tanggungan Negara. Pelaksanaan cuti dapat ditunda dalam jangka waktu tertentu, apabila ada kepentingan dinas yang mendesak. Cuti yang ditangguhkan dalam jangka waktu tertentu tidak dapat ditangguhkan lagi.[14]

2.6.2.5 Definisi Pensiun

Pensiun adalah jaminan hari tua sebagai penghargaan atas jasa-jasa Pegawai Negeri selama bertahun-tahun bekerja dalam dinas pemerintahan. Oleh karena itu pemberhentian sebagai PNS harus dengan sebutan “dengan hormat” dan untuk dapat diberikan pensiun harus memenuhi syarat usia dan masa kerja. Pensiun pegawai dan pensiun janda/duda pegawai diatur dalam Undang-Undang Nomor 11 Tahun 1969.[14]

2.6.2.6 Definisi Promosi

Promosi adalah proses kegiatan pemindahan pegawai atau karyawan dari satu jabatan yang lebih tinggi

serta diikuti oleh tugas, tanggung jawab, dan wewenang yang lebih tinggi dari jabatan yang diduduki sebelumnya.[15]

2.6.3. Fungsi BKD

Fungsi dari Badan Kepegawaian Daerah adalah

a. Penyiapan perumusan kebijakan teknis dibidang manajemen PNS Daerah.
b. Pelayaran penunjang penyelenggaraan pemerintahan daerah di bidang manajemen PNS Daerah.
c. Penyusunan rencana dan program, monitoring evaluasi dan pelaporan manajemen PNS Daerah.
d. Penyiapan penyusunan perundangg-undangan daerah di bidang kepegawaian.
e. Perencanaan dan pengembangan kepegawaian daerah.
f. Penyiapan dan pelaksanaan pengangkatan, kenaikan pangkat, pemindahan dan pemberhentian PNS Daerah.
g. Pelayanan Administrasi Kepegawaian dalam pengangkatan, atau pemindahan dan pemberhentian dalam dan dari jabatan structural funsional.
h. Penyiapan dan penetapan pension Pegawai Negeri Sipil Daerah.
i. Penyiapan penetapan gaji, tunjangan dan kesejahteraan PNS Daerah.
j. Penyelanggaraan administrasi Pegawai Negeri Sipil Daerah.
k. Pengelolaan sistem informasi kepegawaian daerah.
l. Pengelolaan urusan rumah tangga BKD.