BAB II

LANDASAN TEORI

Pada bab ini membahas tentang teori-teori yang mendasari perancangan dan pembuatan dari aplikasi.

2.1. *Fuzzy Logic*

Untuk menghitung gradasi yang tidak terbatas jumlahnya antara benar dan salah, Zadeh mengembangkan ide penggolongan himpunan (*set*) yang ia namakan himpunan *fuzzy* (*fuzzy set*). Tidak seperti logika *boolean* yang menyatakan bahwa suatu pernyataan adalah benar atau salah, *fuzzy logic* dapat membaginya dalam derajat keanggotaan dan derajat kebenaran sehingga suatu pernyataan dapat menjadi sebagian benar dan sebagian salah pada waktu yang sama.

2.1.1. Sejarah fuzzy logic

Fuzzy logic secara resmi diperkenalkan pada tahun 1965, oleh Lotfi Zadeh (*Professor on Systems Theory at University of California, Berkeley*) melalui jurnalnya yang berjudul "*Fuzzy Set*" dalam jurnal *Information and Control.* Dan dalam paper penting lain yang ditulisnya yaitu "*Outline of a New Approach to the Analysis of Complex Systems and Decision Process*", dipublikasikan pada tahun 1973 dalam jurnal *IEEE Transactions on System, Man and Cybernetics,* menjelaskan mengenai pemikiran fuzzy *(fuzzy reasoning).*

Namun Lotfi Zadeh bukanlah orang yang menciptakan fuzzy logic / logika fuzzy, karena logika fuzzy telah ada sejak dulu kala. Bisa kita lihat dari teori dari Arisotle dan George Boole, dimana menjelaskan representasi kebenaran dalam dua

nilai benar / salah, [0,1]. Dan jauh sebelum itu, ahli filsafat Yunani kuno, yaitu Heraclitus dan Anaximander telah mengenal sistem logika ini. (Earl Cox, *Fuzzy Logic for Business and Industry,1995*).

Dan sejak permulaan abad ke-20, Max Black telah menulis mengenai fuzzy set dalam penelitiannya mengenai ketidakjelasan (*vagueness).* Kemudian pertengahan tahun 1930, Jan Lukasiewicz, dimana dikenal sebagai penemu dari notasi tingkah laku *(polish notation),* mengembangkan sistem logika yang memperluas nilai kebenaran untuk semua bilangan riil dari 0 sampai 1. Ia menggunakan himpunan bilangan tersebut mempresentasikan kemungkinan dari pernyataan yang telah diberikan benar atau salah.

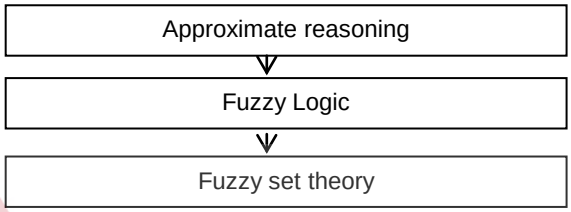
Perkembangan teori Logika Fuzzy telah menarik pakar sistem kendali untuk memanfaatkannya dalam pengendalian suatu sistem dalam bentuk algoritma - algoritma automatik yang dapat dinyatakan, seperti dalam pemakaian pengaturan lalu lintas, sistem transmisi otomatis, alat rumah tangga, industri dan lain-lainnya.

Aplikasi Logika Fuzzy pada industri pertama kali diterapkan setelah tahun 1970 di Eropa. Di Queen Mary College, London, Inggris, Ebrahim Mamdani menggunakan logika fuzzy untuk mengontrol generator uap, dengan menggunakan teknik konvensional.

RWTH University of Aachen, Germany, Hans Zimmermann juga menggunakan logika fuzzy dalam sistem pengambil keputusan (DSS). Dan sampai saat ini Logika Fuzzy masih tetap masih digunakan.

2.1.2. Definisi fuzzy logic

Suatu logika dimana mencoba menggabungkan suatu pertidaksamaan dengan bahasa dengan kejadian alam dengan perhitungan kekuatan komputer untuk menghasilkan kecerdasan tinggi, kuat dan sistem pemikiran yang fleksibel. Namun logika fuzzy, dalam kehidupan sehari-hari, cakupannya benar-benar luas. Lapisan perbedaannya dapat dilihat pada gambar berikut:



Gambar 2.1 Berbagai Aspek dari Fuzzy Logic

Digambarkan yang paling dibawah adalah *Fuzzy set Theory*, yang menggambarkan ilmu mekanik dari bagaimana *fuzzy sets* mengatur dan operasi apa yang diijinkan. Logika Fuzzy itu sendiri adalah proses pembuatan kesimpulan logis dari kumpulan-kumpulan *fuzzy set*. Dalam banyak aplikasi control dan mesin hal ini digambarkan paling atas, karena tak ada teknologi lain yang dibutuhkan.

Dan yang paling atas adalah *approximate reasoning*, suatu kombinasi dari logika matematika dan *heuristic* yang sangat kuat. *Approximate reasoning* merupakan alat yang digunakan oleh *fuzzy expert* dan sistem pendukung keputusan dan termasuk didalamnya penentuan batas *(hedges)* fuzzy set, aturan-aturan *(rules)*, dan bentuk operator.

2.1.3. Kemampuan fuzzy logic

a. Beroperasi tanpa campur tangan manusia secara langsung, tetapi sama efektifitasnya dengan kontroller manusia
b. Mampu menangani sistem-sistem yang kompleks, non linier dan tidak stasioner
c. Strukturnya sederhana dan beroperasi secara *real time*
d. Dapat melengkapi kekurangan dari model *boolean* dalam mempresentasikan hal-hal pada dunia nyata
e. Mampu memenuhi kebutuhan untuk memecahkan masalah yg kompleks dan presisi
f. Kecepatan dalam pengembangan dan kemudahan dalam implementasi

2.1.4. Konsep utama fuzzy logic

A. Prinsip ketidakpastian

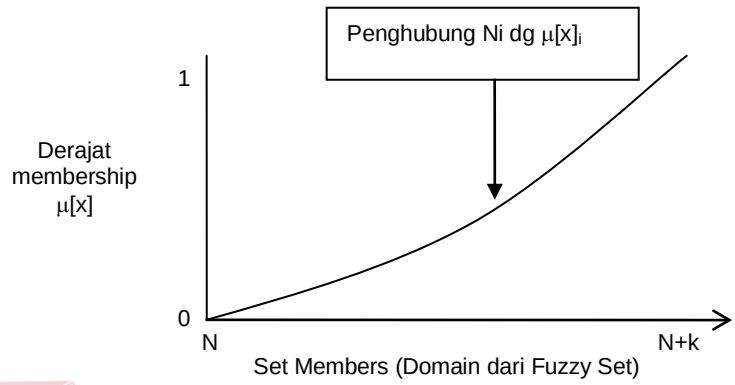
Beberapa ilmu matematika terkadang sulit untuk dipastikan, seperti teori probabilitas, teori informasi, teori fuzzy set. Hal ini bisa diklasifikasikan berdasar tipe ketidakpastian yang dilakukan. Ada beberapa tipe ketidakpastian, dua diantaranya adalah *Stochastic Uncertainty dan Lexical Uncertainty*.

Stochastic Uncertainty berhubungan dengan arah ketidakpastian dari kejadian yang pasti. Sedangkan *Lexical Uncertainty* merupakan ketidakpastian yang diungkapkan oleh kata-kata manusia, seperti “orang tinggi”, “hari yang panas”.

B. Fuzzy sets

Fuzzy set terdiri atas 3 bagian, dimana sumbu horisontal menunjukkan kumpulan member, sumbu vertikal menunjukkan derajat dari membership, dan garis yang

menghubungkan masing-masing titik dari member dengan derajat membership yang tepat.



Gambar 2.2 Fuzzy Sets

C. Membership function

Derajat dimana angka teknis bernilai sesuai konsep bahasa dari kondisi variabel bahasa (*linguistic*) dinamakan sebagai derajat membership. Untuk variabel berlanjut (*continous variable*) derajat ini disebut *Membership function* (MBF).

D. Variabel linguistik

Logika Fuzzy pada dasarnya menitikberatkan pada pengukuran dan penalaran tentang kekaburan atau bentuk fuzzy yang nampak dalam bahasa alami. Dalam logika fuzzy bentuk fuzzy dinyatakan sebagai variabel linguistik (disebut juga variabel fuzzy).

Variabel linguistik adalah bentuk yang digunakan dalam bahasa alami untuk menggambarkan beberapa konsep yang biasanya mempunyai kekaburan atau nilai

fuzzy. Sebagai contoh dalam pernyataan “Jack adalah muda” menyatakan bahwa variabel linguistik umur mempunyai nilai linguistik muda.

Range dari nilai kemungkinan sebuah variabel linguistik disebut semesta pembicaraan dari variabel. Sebagai contoh diberikan range variabel suhu yang digunakan pada rule 1 antara 0-15 derajat. Kata “suhu rendah” anggota dari semesta pembicaraan dari variabel. Ini merupakan himpunan fuzzy.

E. Aturan fuzzy

Aturan dari Sistem Logika Fuzzy (*Fuzzy Logic System*) menggambarkan pengetahuan dari sistem. Mereka menggunakan variabel linguistik sebagai bahasanya, sebagai contoh untuk mengekspresikan strategi control dari sebuah pengontrol logika fuzzy. Menjelaskan aturan fuzzy berarti menunjukkan, bagaimana menghitung dengan konsep linguistik.

2.1.5. Perhitungan fuzzy

Dalam aplikasi kontrol, komputasi fuzzy terdiri atas 3 bagian, yaitu :

A. Fuzzifikasi

Fuzzifikasi berarti menggunakan *Membership function* dari variabel linguistik untuk menghitung masing-masing derajat kondisi validitas dari angka-angka spesifik proses. Fuzzifikasi yang mentransformasi masukan himpunan klasik (*crisp*) ke derajat tertentu yang sesuai dengan aturan besaran fungsi keanggotaan (*Membership function*).

B. Inference

Sistem inferensi fuzzy adalah sistem kerja komputer yang didasarkan pada konsep teori fuzzy, aturan *if-then*, dan logika fuzzy. Struktur dasar dari sistem Inferensi Fuzzy terdiri dari basis aturan yang berisi aturan *if-then*, basis data yang mendefinisikan fungsi keanggotaan dari himpunan fuzzy.

C. Defuzzifikasi

Defuzzifikasi yang mentransformasi hasil fuzzy ke bentuk keluaran yang crisp. *Membership function* digunakan dalam menterjemahkan keluaran fuzzy ke bentuk keluaran *crisp*. Terjemahan kembali disini dapat menggunakan beberapa metode, dimana diantaranya adalah CoM (*Center of Maximum*) / CoG (*Center of Gravity*), CoA (*Center of Area*), MoM (*Mean of Maximum*), MoM BSUM (*Mean of Maximum Bounded Sum*).

2.1.6. Himpunan fuzzy

Teori himpunan tradisional menggambarkan dunia sebagai hitam dan putih. Ini berarti sebuah obyek berada didalam atau diluar himpunan yang diberikan. Dalam teori himpunan tradisional untuk anggota diberi nilai 1 dan untuk bukan anggota diberi nilai 0; ini disebut himpunan *crisp*. Sebagai contoh anggota himpunan orang muda dapat berisi hanya orang yang berumur kurang dari 10. Penggunaan interpretasi ini pada seseorang yang berulang tahun ke-11, maka orang tersebut bukan anggota himpunan orang muda.

Himpunan fuzzy memberikan nilai keanggotaan antara 0 dan 1 yang menggambarkan secara lebih alami sebuah kumpulan anggota dengan himpunan, Sebagai contoh, jika seorang berumur 5 tahun dapat diberikan nilai keanggotaan 0.9

atau jika umurnya 13 tahun nilai keanggotaannya 0.1. Dalam contoh ini “umur” adalah variabel linguistik dan “muda” adalah salah satu himpunan fuzzy.

Definisi himpunan Fuzzy :

Misalkan X semesta pembicaraan, dengan elemen dari X dinotasikan x. Sebuah himpunan fuzzy A dari X dikarakteristikkan dengan fungsi keanggotaan

$\mu \wedge (x) : X \rightarrow [0,1]$

Pada logika fuzzy, kejadian atau elemen x diberikan nilai keanggotaan dengan fungsi keanggotaan μ. Nilai ini mempresentasikan derajat keanggotaan elemen x pada himpunan fuzzy A.

$\mu_{\wedge}(x)$ = Degree($x \in A$)

Nilai keanggotaan dari x berada pada interval :

$0 \leq \mu_{\wedge}(x) \leq 1$

Himpunan fuzzy adalah perluasan dari teori himpunan tradisional. Himpunan fuzzy menyamakan konsep keanggotaan dengan menggunakan fungsi keanggotaan μ yang menghasilka nilai antara 0 dan 1 yang mempresentasikan derajat keanggotaan obyek x pada himpunan A.

2.1.7. Membentuk himpunan fuzzy

Untuk mempresentasikan himpunan fuzzy dalam komputer perlu didefinisikan fungsi keanggotaannya. Sebagai contoh : orang tinggi. Dapat dinyatakan pada setiap individu, pada tingkatan mana bahwa mereka yakin seseorang itu dikatakan tinggi. Setelah mengumpulkan jawaban untuk interval ukuran tinggi, dapat disajikan tingkat rata-rata untuk menghasilkan suatu himpunan fuzzy dari orang-orang yang tinggi.

Fungsi ini dapat digunakan sebagai suatu keyakinan (nilai keanggotaan). Bagi individu yang menjadi anggota himpunan fuzzy dari orang tinggi.

Dengan membentuk fuzzy subset untuk berbagai bentuk fuzzy, dapat dianggap nilai keanggotaan dari obyek yang diberikan pada setiap himpunan. Pendekatan lain yang sering ditemukan pada praktek untuk membentuk himpunan fuzzy sangat berhubungan dengan interpretasi dari seorang ahli. Seperti teknik pengumpulan data, dapat ditanyakan pada pakar untuk kepercayaannya bahwa berbagai obyek merupakan bagian himpunan yang diberikan.

2.1.8. Batasan

Dalam pembicaraan normal, manusia mungkin menambahkan kekaburan untuk memberikan pernyataan dengan menggunakan kata keterangan seperti sangat, agak. Kata keterangan adalah sebuah kata yang memodifikasi kata benda, kata sifat, kata keterangan lain, atau keseluruhan kalimat. Sebagai contoh, kata keterangan memodifikasi kata sifat, “orang itu sangat tinggi”.

Sebuah hedges memodifikasi himpunan fuzzy yang sudah ada secara matematis untuk menghitung beberapa kata keterangan yang ditambahkan.

2.1.9. Operasi himpunan fuzzy

Terdapat 3 operasi dalam himpunan fuzzy, yaitu :

A. Irisan

Dalam teori himpunan klasik, irisan dari dua himpunan berisi elemen-elemen yang sama dari keduanya. Dalam himpunan fuzzy, sebuah elemen mungkin sebagian dalam kedua himpunan. Oleh karena itu ketika mengingat irisan dari

kedua himpunan, tidak dapat dikatakan bahwa sebuah elemen adalah lebih mungkin menjadi dalam irisan daripada dalam suatu himpunan asli.

B. Gabungan

Cara kedua dari penggabungan himpunan fuzzy adalah gabungannya. Penggabungan dari dua himpunan adalah terdiri dari dua himpunan adalah terdiri dari elemen-elemen yang menjadi satu atau dua himpunan. Dalam situasi ini anggota dari gabungan tidak dapat mempunyai nilai keanggotaan yang kurang dari nilai keanggotaan yang lain dari himpunan aslinya.

C. Komplemen

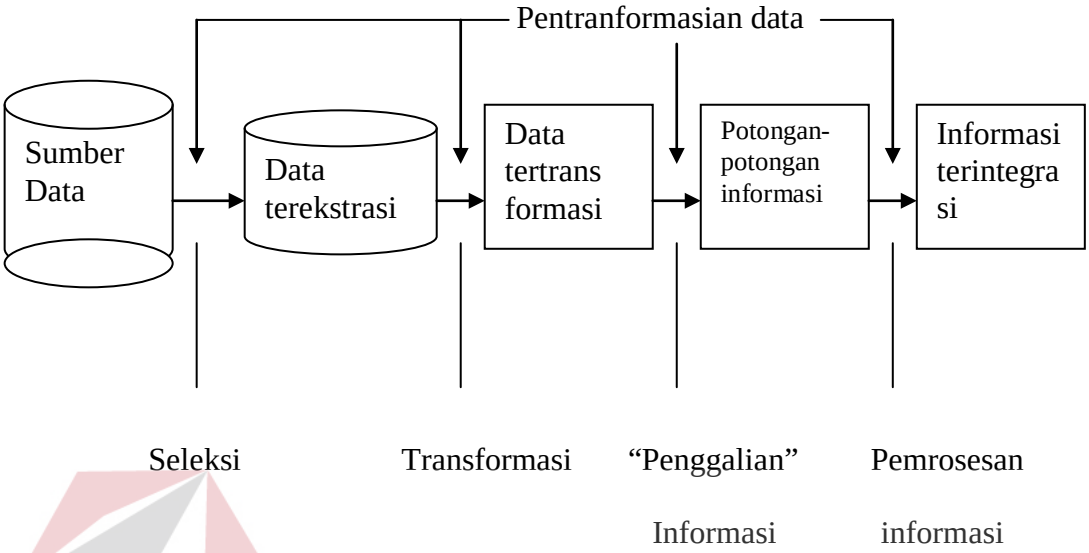
Komplemen dari himpunan fuzzy A dinotasikan dengan (~A) dinyatakan dengan persamaan sebagai berikut :

$\mu_{\sim A}(x) = 1 - \mu_A(x)$

2.2. Data Mining

Proses pengektrasian informasi yang sebelumnya tidak diketahui, merupakan hal yang penting dari suatu sumber basis data, dan hal ini dapat dirumuskan untuk kepentingan bisnis. Data Mining adalah tentang sebuah penemuan hal-hal yang benar-benar nyata (discovery of Facts). Sesuatu yang awalnya tidak saling berhubungan, tetapi jika dilihat dari sudut yang lebih luas dengan menggunakan kepandaian manusia dan teknologi, akan memiliki informasi yang berguna (Berson dkk, 2001).

2.2.1. Proses data mining :



Gambar 2.3 Proses Data mining

Diagram diatas adalah sebuah gambar proses Data Mining. Diawali dari sebuah sumber data lalu dilakukan seleksi-seleksi untuk memilih data apa yang akan dianalisa. Hasil seleksi Data Mining kemudian diubah menjadi bentuk algoritma Data Mining dapat dilakukan. Hasilnya adalah suatu informasi penting tentang karakteristik dari data itu sendiri yang akan sangat membantu para pelaku pasar untuk melakukan tindakan.

Data mining relevan untuk diimplementasikan diberbagai bidang mulai dari perdagangan / pemasaran, keuangan, pelayanan kesehatan hingga telekomunikasi. Ada 2 tahap proses data mining yang disesuaikan intuk pencarian informasi :

1. Pencarian awal dilakukan terhadap data untuk menghasilkan informasi yang benar-benar dibutuhkan.

2. Mengarahkan perhatian pada detil untuk memberikan interpretasi yang lebih jelas terhadap informasi yang dihasilkan.

Konsep data mining menyediakan kemampuan untuk menganalisa dan memonitor kecenderungan dan pola pembelian pelanggan. Sehingga dapat mendukung seorang pakar dalam pengambilan keputusan. Data mining melibatkan proses yang komplek dan biaya yang tidak sedikit. Agar penggunaan data mining berlangsung efektif dan efisien harus disiapkan terlebih dahulu lingkup informasi dan data apa saja yang patut dilibatkan.

Keuntungan data mining :

1. Hasilnya merupakan kumpulan alamiah data berdasarkan kemiripan
2. Jumlah cluster / kelompok yang ingin dihasilkan dapat ditentukan sendiri (konsep hirarki)

2.2.2. Pengelompokan

Metode ini pada dasarnya melakukan segmentasi / pengelompokan suatu populasi data yang heterogen menjadi beberapa subgrup atau cluster yang homogen. Metode ini dikategorikan kedalam teknik *undirect knowledge* dan *unsupervised learning* karena tidak membutuhkan proses pelatihan untuk mengklasifikasi awal data dalam masing-masing grup atau cluster. Selain itu tidak ada perbedaan perlakukan antara variable-variabel dependen dalam studi kasus ini satu sama lain dan variable-variabel independen. Record data dikelompokkan berdasarkan kemiripan data antara satu record dengan record yang lain.

Pengguna harus mengartikan sendiri cluster-cluster yang dihasilkan, karena cluster-cluster tersebut belum dipridiksikan sebelumnya oleh pengguna, oleh karena

itu output metode ini sering kali dijadikan input metode data mining atau pemodelan data yang lain untuk menghasilkan informasi yang benar-benar siap dipakai untuk analisa data.

2.2.3. Pengklasifikasian

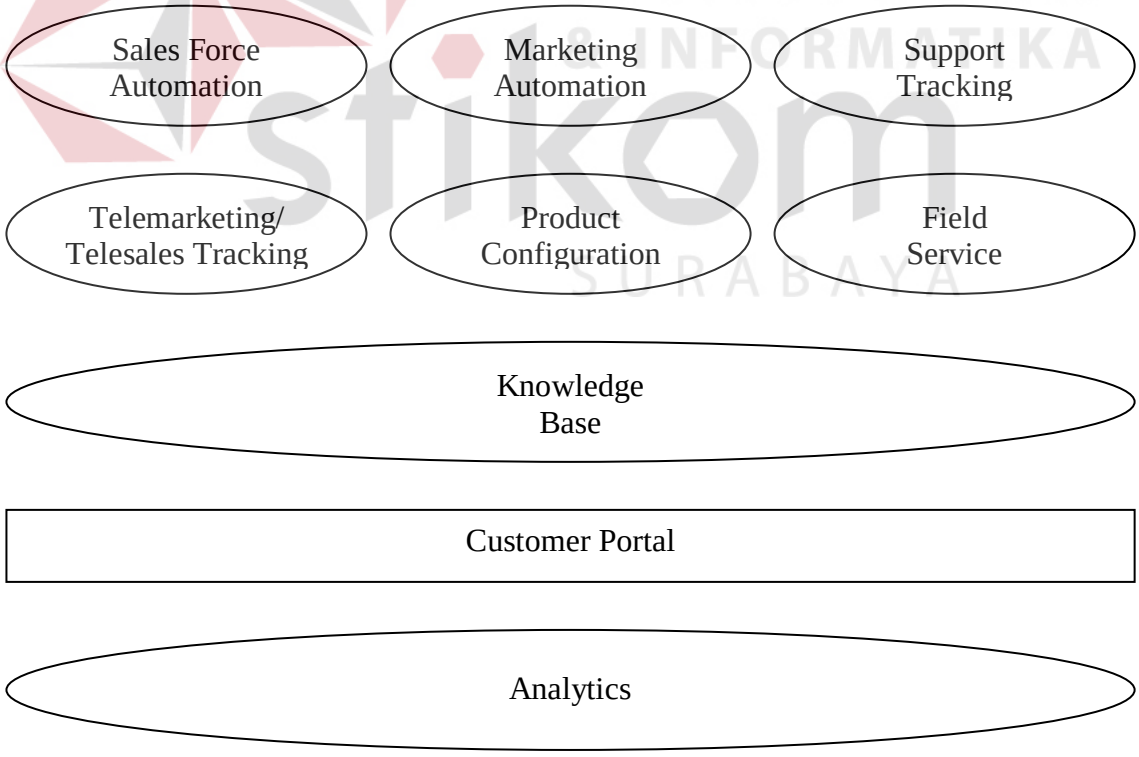
Klasifikasi merupakan bentuk umum dari data mining. Klasifikasi menyediakan cara penyediaan obyek dan menggabungkannya menjadi kelas-kelas. Untuk lebih mudahnya obyek yang kita klasifikasikan secara umum direpresentasikan dalam satu record didalam suatu database dan pengklasifikasian menyediakan fasilitas untuk mengupdate setiap record dengan memberikan kode untuk setiap jenis record yang sama. Walaupun antara clustering dan classification memiliki kemiripan (kedua-duanya sama-sama melakukan pengelompokan), namun pada dasarnya keduanya sangatlah berbeda. Kenyataannya kita seringkali harus mempergunakan algoritma clustering dan algoritma classification secara bersamaan. Pertama-tama kita harus mempergunakan algoritma clustering untuk menemukan cara terbaik dalam menentukan kemiripan karakteristik kelompok-kelompok data yang ada. Kemudian kelompok-kelompok tersebut kita jadikan kelas-kelas. Data yang ada dimasukkan dengan kelompok-kelompoknya. Akhirnya kita gunakan algoritma classification untuk menentukan rule atau rumus-rumus untuk menentukan suatu data baru masuk ke kelas yang mana.

2.3. Customer Relationship Management

Menurut Tourniaire dan Francoise (1993). *Customer Relationship Management* (CRM) digunakan untuk menggabungkan tiga elemen, yaitu :

1. Masukkan dari CRM yang terdiri dari pelanggan dan sales.
2. Peralatan yang digunakan oleh *Sales Force Automation* (SFA).
3. Sesuatu yang berada diantara keduanya, biasanya proses yang terlibat dalam hubungan manajemen dengan pelanggan.

CRM mempunyai kemampuan dalam menangani kekacauan *field*, dan beberapa *vendor* yang mengambil tindakan hanya sedikit memperbaiki teka-teki yang mereka sebut CRM *vendor*, hal ini menyulitkan untuk menyaring tuntutan yang terus menerus. Dalam penjumlahan, beberapa back-office-system merupakan rangkaian awal hubungan dengan pelanggan, dan keistimewaan yang lain yaitu menutup CRM fungsional. Jadi, batasan bukan merupakan potongan keberhasilan mereka. Gambar dibawah merupakan hasil survei dari fungsional klasik untuk sistem CRM.



Gambar2.4 Elemen dari CRM

2.3.1. Sales force automation

Sales force automation (SFA) disebut juga bintang dari tanah lapang, merupakan salah satu driver dari *field* CRM, dan yang satu lagi adalah wujud dari *support tracking.* Karakteristik unik dari SFA adalah keperluan untuk membantu bergerak, memutuskan hubungan pemakai. Dasar dari kebutuhan penjual yaitu pengambilan informasi akuntansi yang sesuai dengan komputer mereka, memperbaharui jika pergi dari kantor dan kemudian memuat kembali di dalam mengambil bagian sistem. Ini disebut juga kekuatan kebersamaan dalam pengambilan dan pemuatan kejadian yang dapat terjadi dengan cepat serta memelihara atau mempertahankan data secara konsisten jika terjadi pembaharuan yang berkali-kali.

2.3.2. Telemarketing and telesales tracking

Walaupun *Telemarketing and telesales tracking* bisa menganggap bagian dari SFA, peralatan untuk operasi *Telemarketing and telesales* mempunyai sedikit perbedaan kerena *telemarketers* sering bekerja dari naskah tulisan atau berakhir pada banyak struktur lingkungan sekitar. Mereka tidak membutuhkan bergerak atau berpindah untuk memutuskan hubungan fungsional. Dalam beberapa jalan, mereka perlu untuk menutup itu dari *support tracking* dan tentu saja vendor menutup pusat hubungan yang dapat melayani kedua-duanya di dalam batas dan di luar pusat hubungan. Bagian fungsional dari aplikasi pusat hubungan adalah *Computer telephony integration* (CTI), dengan memberikan informasi dalam mengambil bagian diantara sistem telepon dan aplikasi CRM. Keistimewaan dari aplikasi CTI yaitu *routing calls*, yang merupakan salah satu dasar dari nomer telepon yang masuk atau pada informasi yang masuk oleh pelanggan dalam sistem telepon yang ditunjukkan

oleh menu pilihan dan kemampuan untuk menunjukkan layar yang berisi informasi panggilan. CTI dapat juga memanipulasi dasar sistem telepon dalam database CRM, untuk langsung memanggil nomer telepon yang tertera pada layar.

2.3.3. Product configuration

Product configuration atau bentuk wujud dari produk merupakan peralatan yang memberi pemakai kebiasaan yang komplek untuk meneliti keperluan dari produk tersebut. Bentuk wujud dari produk digunakan untuk pekerjaan yang membawa urusan di luar hubungan dengan pelanggan, tetapi sekarang menjadi bagian dari beberapa CRM yang baik.

2.3.4. Marketing automation

Marketing automation sering disebut dengan manajemen kampanye, *Marketing automation* memberikan desain, pelaksanaan dan pengaturan dalam kampanye. Tergantung pengalaman dari peralatan, kampanye mungkin menggunakan bermacam-macam media dan termasuk *segmentation* dan *list management*.

2.3.5. Support tracking

Dengan SFA, *support tracking* merupakan satu lagi sejarah yang penting dari CRM. Dasar *support tracking* yaitu mengolah data-data sehingga dapat digunakan sebagai dasar dalam pengambilan keputusan. CTI dan sistem komunikasi elektronik sering digunakan untuk mengambil tindakan.

2.3.6. Field service

Field service mempunyai perbedaan keperluan dari pada pelayanan yang mengambil tindakan dari support center.

2.3.7. Knowledge base

Knowledge base fungsional sangat berguna dalam semua area dari fokus fungsi pelanggan. Mereka sesungguhnya mempunyai dua perbedaan tipe dari fungsional. Yang pertama kemampuan untuk menunjukkan *knowledge base* kepada pengguna melalui bermacam-macam kemampuan penyelidikan dan yang kedua lebih penting dalam beberapa jalan walaupun tanpa diskusi yaitu kemampuan untuk mendukung kreasi dan mempertahankan dokumen.

2.3.8. Customer portal

Web-based akses pelanggan untuk sistem CRM sekarang ini mutlak diperlukan. Dasar *customer portal* memberikan akses ke *knowledge base* dan ke sistem *request-tracking*, tetapi beberapa fungsi yang berpengalaman merupakan harapan termasuk cabang lokasi, pengambilan elektronik, dan komunikasi *online*. C*ustomer portal* dan fungsional sekitarnya suatu saat akan disebut sebagai eCRM dan subsistemnya disebut *e-sales, e-marketing atau e-support*.

2.3.9. Analytics

Satu manfaat dari CRM adalah mengembangkan kemampuan untuk memperlihatkan dan menganalisa aktivitas pelanggan. Dari dulu orang miskin sangat terbatas dalam menggunakan CRM, analisa kemudian menjadi bisnis besar dan sering

menjual secara terpisah dengan pilihan harga tambahan. Walaupun pemberitahuan sangat penting, kunci utama kesuksesan untuk analisa adalah mempunyai data yang baik, jadi pastikan bahwa data yang anda punyai merupakan gambaran dari sistem.

2.4. Analisa dan Perancangan Sistem

2.4.1. Penggunaan DFD

Meskipun suatu analisa yang disebut dengan DFD mempunyai struktur tersendiri, namun sistem analisa dapat meletakkan secara bersamaan sebuah gambar yang merepresentasikan seluruh proses-proses data dalam sebuah organisasi. Pendekatan data *flow* menitik beratkan pada logika yang tersirat dari suatu sistem (Kendall, and Kendall, 2002).

Dengan menggunakan kombinasi simbol, sistem analisa dapat membuat sebuah gambaran dari suatu proses yang sebenarnya dengan menggunakan dokumen sistem.

A. Keuntungan pembuatan data flow

Data *flow* mempunyai lima keuntungan utama dari penjelasan-penjelasan jalannya data dalam sistem, yaitu :

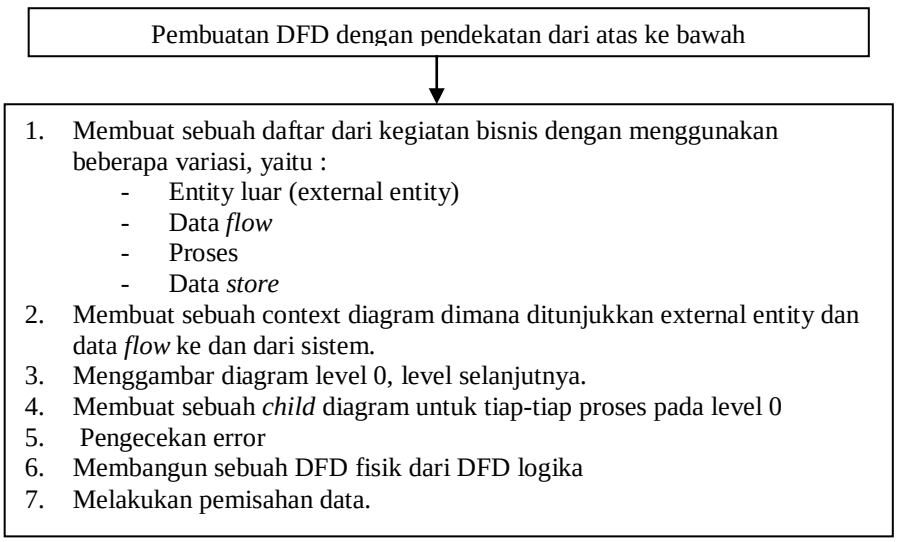
1. Kebebasan yang berasal dari kepercayaan untuk mengimplementasikan secara benar teknik sistem dari suatu sistem yang baru.
2. Memberikan pengertian dari hubungan sistem-sistem dan subsistem yang ada.
3. Komunikasi mengenai pengetahuan sistem bagi *user* melalui DFD
4. Analisa dari sebuah usulan sistem untuk menentukan jika data dan proses-proses yang ada dapat didefinisikan secara mudah.

5. Penggunaan data *flow* merupakan keuntungan tambahan yang dapat digunakan sebagai latihan bagi sistem analis, kesempatan sistem analis menjadi lebih baik untuk mengerti tentang hubungan sistem dan subsistem yang ada didalamnya.

Keuntungan dari kelima penggunaan data *flow* tersebut dapat digunakan sebagai *tools* yang interaktif dengan *user*. Hal yang menarik dalam penggunaan DFD adalah ditunjukannya kepada *user* gambaran-gambaran secara lengkap dari sistem. *User* dapat menanyakan guna memberikan komentar pada konsep, sistem analis dapat merubah sistem berdasarkan keinginan *user*. Keuntungan terakhir dari penggunaan DFD adalah dapat mengikuti sistem analis untuk mendeskripsikan komponen-komponen yang digunakan dalam suatu diagram. Analisa dapat ditampilkan untuk menjamin bahwa semua *output* mempunyai isi atau memperoleh data inputan dari prosesnya.

B. Pembuatan DFD

DFD dapat dan harus digambarkan secara sistematis . Pertama, dibutuhkan sistem analis untuk mengkonsep data *flow*, dari atas ke bawah seperti ditunjukkan pada gambar berikut:



Gambar 2.5 Pembuatan DFD

Untuk memulai sebuah DFD dari suatu sistem biasanya dituangkan dalam sebuah daftar dengan empat kategori yaitu entity luar, arus data, proses, dan penyimpanan data. Daftar ini akan membantu menentukan batasan-batasan dari suatu sistem yang akan digambarkan. Pada dasarnya daftar itu berisi elemen-elemen data yang dikarang. Elemen-elemen tersebut terdiri dari

a. Pembuatan context diagram

Context diagram adalah level yang tertinggi dalam sebuah DFD dan hanya berisi satu proses serta merupakan representasi dari sebuah sistem. Proses dimulai dengan penomeran ke-0 dan untuk seluruh entity luar akan ditunjukkan dalam context diagram yang sama seperti data awal yang dikirim dari entity luar. Context diagram tidak berisi penyimpanan data.

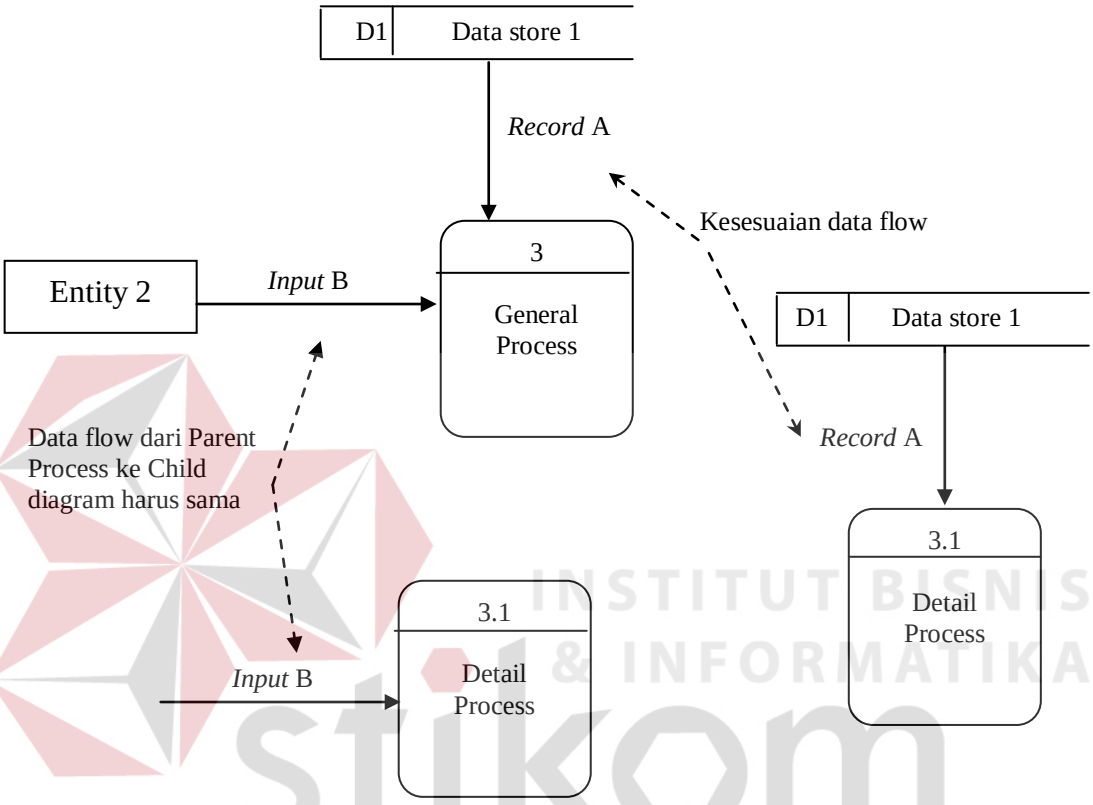
b. Pembuatan diagram level 0 serta level berikutnya

Diagram level 0 dihasilkan oleh context diagram dan berisi proses-proses. Pengisian proses-proses yang berlebihan pada level ini akan menghasilkan sebuah diagram yang salah, sehingga sulit umtuk dimengerti . Masing-masing proses diberikan penomoran dengan sebuah bentuk integer. Umumnya dimulai dari kiri atas dan penyelesaiannya di kanan bawah dalam sebuah bentuk diagram.

c. Pembuatan *child* diagram

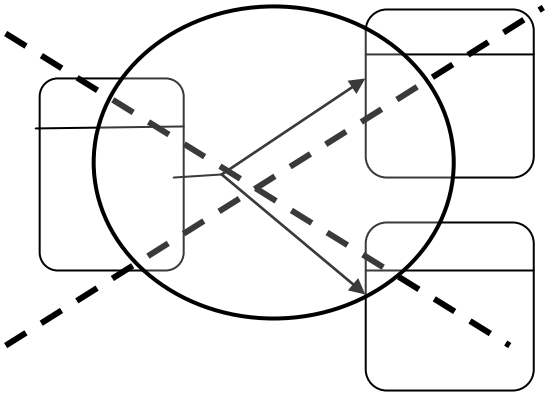
Child diagram diberikan nomor yang sama seperti proses diatasnya (parent proses) dalam diagram level 0. Contohnya, proses 3 harus diturunkan ke diagram 3, proses pada *child* diagram menggunakan penomoran unik untuk masing-masing proses dengan mengikuti penomoran proses diatasnya . Contohnya, dalam diagram 3 proses-proses diberikan nomor 3.1, 3.2, 3.3 dan seterusnya. Konversi ini diikuti oleh analis sistem untuk menelusuri seri-seri dari proses-proses yang dikeluarkan

oleh beberapa level, jika pada proses diagram level 0 digambarkan sebagai 1, 2, , dan 3 maka *child* diagram-diagramnya adalah 1, 2, dan 3 pada level yang sama. Ilustri level detil dengan sebuah *child* DFD dapat ditunjukkan pada gambar :



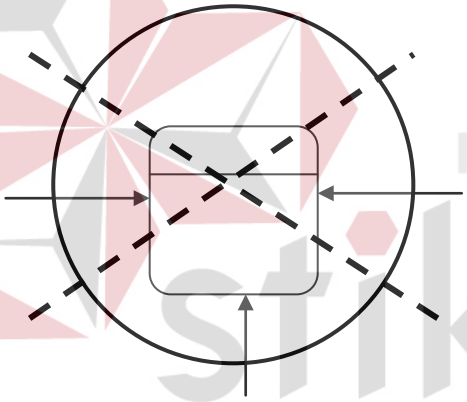
Gambar 2.6 Ilustrasi level detil dengan sebuah *child*DFD

d. Pengecekan kesalahan-kesalahan pada diagram digunakan untuk melihat kesalahan-kesalahan yang terdapat pada sebuah DFD. Beberapa kesalahan-kesalahan yang umum terjadi ketika penggambaran / pembuatan DFD, ditunjukkan pada gambar berikut, adalah :



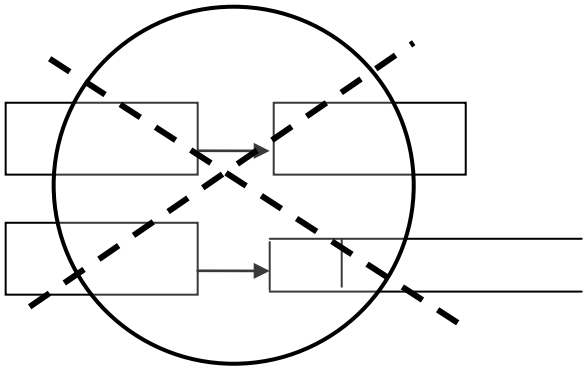
Sebuah data flow tidak diperbolehkan mempunyai percabangan / memisahkan diri (flow) ke dalam dua atau lebih data flow yang berbeda

Gambar 2.7 Pengecekan kesalahan 1



Sebuah proses harus mempunyai minimal satu *input*an data flow dan satu *output* data flow

Gambar 2.8 Pengecekan kesalahan 2

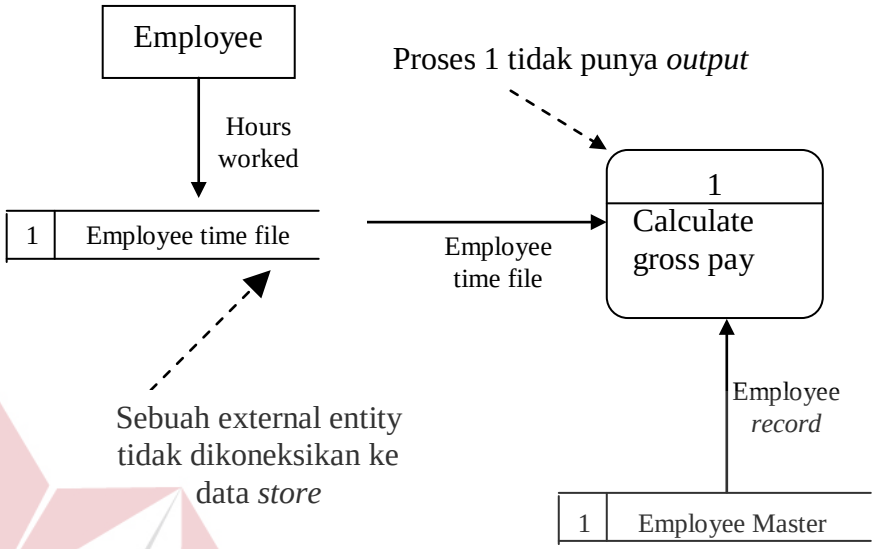


Semua data flow salah satunya harus berasal atau berakhir pada sebuah proses

Gambar 2.9 Pengecekan kesalahan 3

1. Lupa untuk meng*input*kan sebuah arus data atau arah panah langsung. Sebagai contoh adalah penggambaran proses yang menunjukkan sebuah data *flow* seperti *input* atau seperti *output*. Tiap-tiap proses pengubahan data harus menerima *input* dan *output*. Tipe kesalahan ini terjadi ketika sistem analis lupa memasukkan sebuah data *flow* atau meletakkan sebuah arah panah ditempat yang salah.
2. Hubungan penyimpanan data dan entity luar secara langsung satu sama lain. Data *store* dan entity tidak mungkin dikoneksikan satu sama lain ; data *store* dan entity luar harus dikoneksikan melalui sebuah proses.
3. Kesalahan penamaan (label) pada proses-proses atau data *flow*. Pengecekan DFD untuk memastikan bahwa tiap-tiap objek atau data *flow* telah diberikan label. Sebuah proses haruslah di indikasikan seperti nama dari sistem atau menggunakan format kata kerja-kata benda. Tiap data *flow* haruslah dideskripsikan dengan sebuah kata benda.
4. Memasukkan lebih dari sembilan proses dalam sebuah DFD. Memiliki banyak proses akan mengakibatkan kekacauan pada diagram sehingga dapat menyebabkan kebingungan dalam pembacaan sebuah proses dan akan menghalangi tingkat komunikasi. Jika lebih dari sembilan proses dalam sebuah sistem, maka beberapa grup dalam proses dilakukan bersama-sama ke dalam sebuah sub sistem dan meletakkannya dalam sebuah *child* diagram.
5. Menghilangkan suatu arus data. Pengujian dari suatu diagram yang menunjukkan garis / arah (*flow*), dimana untuk setiap proses data *flow* hanya mempunyai *input* data, *output* kecuali dalam kasus dari detil (*child*). Setiap *child* data dari DFD,

arah arus data seringkali digambarkan untuk mengidentifikasikan bahwa diagram tersebut kehilangan data *flow*. Seperti di tunjukkan pada gambar :



Gambar 2.10 Kehilangan Data Flow Pada DFD

6. Membuat ketidaksesuiaan komposisi dalam *child* diagram , dimana tiap *child* diagram harus mempunyai *input* dan *output* arus data yang sama seperti proses dilevel atasnya (parent proses). Pengecualian untuk rule ini adalah kurangnya *output*, seperti kesalahan garis yang ada didalam *child* diagram.

C. Perbedaan DFD

Pada tabel berikut digambarkan perbedaan antara DFD logika dan DFD fisik

Tabel 2.1 Perbedaan DFD Logika Dan Fisik

Disain	**Logika**	**Fisik**
Gambaran model	Operasi-operasi bisnis	Bagaimana sistem akan diimplementasikan (atau bagaimana sistem dijalankan)
Apa yang ditampilkan oleh proses	Aktivitas bisnis	Program-program, modul program, dan prosedur-prosedur manual
Apa yang ditampilkan oleh data *store*	Koleksi-koleksi dari data yang dikesampingkan dari bagaimana data tersebut di simpan	File-file fisik dan database-database dari file-file manual
Kontrol system	Menunjukkan kontrol-kontrol bisnis	Menunjukkan kontrol-kontrol untuk validasi *input* data, untuk memperoleh sebuah *record* , untuk memastikan kesuksesan proses dan untuk keamanan sistem

2.4.2 Desain input

Bentuk-bentuk desain secara khusus mungkin dapat digunakan jika suatu analisa sistem dapat disesuaikan dengan bentuk desain secara lengkap dan bermanfaat. Hal itu juga penting untuk mengenalkan secara dini desain yang digunakan, arus data atau bentuk-bentuk yang tidak digunakan pada sumber-sumber suatu organisasi dan itu harus dihilangkan. Untuk mendesain form-form yang baik harus diterapkan antara lain :

1. Membuat form-form tersebut mudah dalam pengisiannya.
2. Memastikan form-form tersebut sesuai dengan tujuan untuk masing-masing desain.
3. Desain form digunakan untuk menjamin kelengkapannya.
4. Mempertahankan form-form yang menarik.

2.4.3. Desain output

Output merupakan sebuah informasi yang dikirim kepada *user* melalui suatu sistem informasi (seperti : Internet, Extranet, dan WWW). Beberapa data yang dibutuhkan diproses secara teliti sebelum dinyatakan layak sebagai sebuah *output*, penyimpanan data lain, dan ketika data-data tersebut dibutuhkan kembali, data-data tersebut berupa *output* yang membutuhkan proses yang sedikit. *Output* dapat diambil dari beberapa bentuk yaitu : bentuk laporan yang dihasilkan printer dan tampilan layer computer, mikrofon (suara).

Oleh sebab itu, penggunaan *output* sangat penting guna menjamin pemakaian dan penerimaan dari suatu sistem informasi. Ada beberaopa objek dimana analis

mencoba untuk mencapai suatu desain output yang tepat. Objek-objek tersebut antara lain :

a. Desain *output* untuk melayani sebuah tujuan tertentu.
b. Pembuatan *output* yang disesuaikan bagi kebutuhan user.
c. Pengiriman sejumlah *output*
d. Pengelolaan distribusi *output*
e. Pengelolaan waktu *output*
f. Penyesuain metode ouput yang paling efektif.

2.4.4. Desain database

A. Bentuk database dan file

Ada dua bentuk pendekatan pada proses penyimpanan data dalam sebuah sistem komputer. Metode pertama adalah untuk menyimpan file-file tunggal, masing-masing dengan bentuk unik untuk berbagai macam aplikasi. Pendekatan yang kedua adalah untuk menyimpan data dalam sebuah sistem komputer dengan melibatkan pembuatan sebuah *database*. Sebuah *database* merupakan sebuah pendefinisian normal dan merupakan pusat penyimpanan dari data tertentu yang digunakan dalam beberapa aplikasi yang berbeda.

B. Database

Database bukan hanya merupakan sebuah koleksi dari suatu file-file. Meskipun, sebuah *database* merupakan sebuah pusat sumber data yang disimpan oleh beberapa *user* dari sebuah aplikasi-aplikai yang bervariasi. Inti dari sebuah *database*

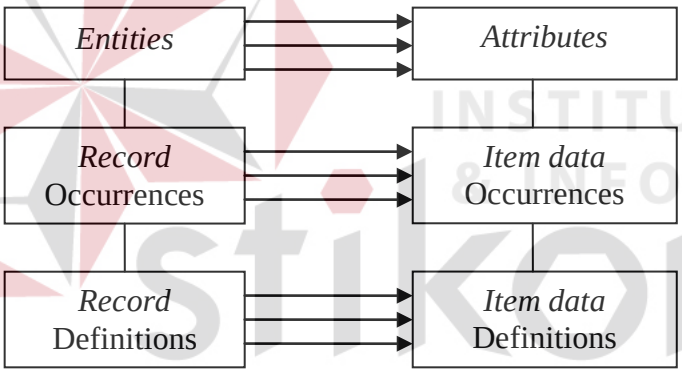
adalah DBMS (*Database Management* Sistem), dimana diikuti dengan kreasi, modifikasi, dan perubahan (*update*) dari *database*.

C. Konsep-konsep data

Hal ini sangat penting untuk dimengerti bagaimana data dipresentasikan kembali sebelum memutuskan penggunaan *file* atau *database*.

1. Realita data dan Meta data

Hubungan antara realita, data dan meta data yaitu didalamnya terdapat entity dan attribut (realita), *record* dan *item data* (data), definisi *record* dan definisi *item data* (meta data). Ditunjukkan pada gambar berikut :



Gambar 2.11 Hubungan Antara Realita, Data, dan Meta data

dimana :

- Entity → Merupakan beberapa obyek atau kejadian tentang dimana dapat mencocokkan koleksi data sebagai sebuah entity. Entity juga dapat berupa sebuah kejadian atau unit dari satu waktu.
- *Relationship*→ Merupakan hubungan antara entity, adapun tipe-tipe dari *relationship* antara lain (1) one to one (1:1), (2) one to many (1:M), dan (3) many to many (M:N).

- Attribut → Merupakan sebuah karakteristik dari sebuah entity.
- *Record* → Merupakan suatu kumpulan dari item-item data yang secara umum merupakan penjelasan umum dari entity.

2. Meta data

Meta data merupakan data dimana dijelaskan tentang data *file* atau *database*. Meta data menjelaskan pemberian nama dan menunjukkan panjang dari masing-masing item data. Meta data juga menjelaskan panjang dan komposisi dari tiap-tiap *record*.

3 Kelompok *file*

Sebuah *file* berisi grup-grup dari *record* yang digunakan untuk melengkapi informasi untuk suatu operasi-operasi, perencanaan, manajemen, dan pembuatan keputusan. Tipe-tipe dari *file* yang digunakan antara lain :

a. *File* master : berisi *record-record* dari sebuah kelompok entity.

b. *File* tabel : berisi data yang digunakan.

c. *File-file* transaksi : digunakan untuk mengisi perubahan (*update*) sebuah *file* master dan laporan-laporan.

d. *Work file* : digunakan untuk menjalankan program agar lebih efektif.

e. *File* laporan : Memudahkan untuk menjalankan program (ketika tidak ada printer).

D. Kumpulan database

1. Data fisik dan logika (*Physical and Logical*)
2. *Hierachical*
3. Struktur jaringan data
4. Struktur hubungan data

E. Normalisasi

Normalisasi merupakan perubahan dari *user* yang ditampilkan secara lengkap dan simpanan data untuk ukuran dari yang terkecil, serta merupakan struktur-struktur data yang stabil. Normalisasi dibutuhkan untuk mengorganisir data dan menghindari redudansi data (data *double*).

Ada tiga bentuk normalisasi, yaitu :

1. *First Normal Form* (1NF)

   Langkah pertama ini terdapat pada sebuah relasi normalisasi yang digunakan untuk menghilangkan (menghapus) grup-grup yang berulang.

2. *Second Normal Form* (2NF)

   Pada bentuk normal yang kedua, seluruh attribut yang ada akan difungsikan tergantung pada PK (*Primary Key*).

3. *Third Normal Form* (3NF)

   Sebuah relasi penormalisasian yang ketiga adalah jika seluruh yang bukan kunci (PK) dari semua attribut seluruhnya difungsikan bergantung pada PK dan attribut tersebut bukan transitif ketergantungan (tanpa kunci).

2.4.5. Desain user interface

Bagaimanapun baik atau buruknya suatu tampilan, hal itu berpengaruh pada keberadaan representasi sebuah sistem. Desain terdiri dari :

a. Tipe-Tipe *User Interface*

Beberapa peebedaan terdiri dari penjelasan *User Interface*, bahasa yang digunakan pada suatu tampilan, jawaban dan pertanyaan dari suatu tampilan, menu-menu, tampilan bahasa yang digunakan, tampilan bentuk isian, GUIs (*Graphical User Interface*) dan variasi dari web *Interface* (untuk program internet).

b. Tampilan Bahasa

Tampilan bahasa sering kali digunakan unruk percobaan bagi *user* guna dapat berinteraksi dengan Komputer.

c. Tampilan Tanya Jawab

Dalam tampilan tanya jawab, tampilan computer merupakan sebuah pertanyaan bagi user, untuk berinteraksi user memasukkan sebuah pertanyaan (melalui tombol keyboard atau mouse) dan computer memberikan aksinya pada suatu informasi output.

d. Menu-Menu

Sebuah tampilan menu menyediakan tempat bagi user untuk melihat nama-nama dari daftar pada form tampilan.

e. Tampilan Bentuk Isian / Bentuk *Input* dan *Output*

Tampilan bentuk isian terdiri dari bentuk-bentuk dalam layer atau bentuk form yang menampilkan isian-isian *field item* data atau parameter-parameter yang dibutuhkan untuk dikomunikasikan oleh *user*.

f. Tampilan Bahasa Perintah

Sebuah tampilan perintah bahasa mengizinkan *user* untuk mengontrol sebuah aplikasi dengan sebuah deretan tombol kunci dan perintah-perintah. Bahasa perintah digunakan untuk memanipulasi komputer, sama seperti *tools* bagi *user* untuk mengontrol dialog.