BAB II

TINJAUAN PUSTAKA

II.1. Perancangan

Menurut Al-Bahra (2005:51) yang terdapat dalam buku yang berjudul Analisis dan Desain Sistem Informasi, menjelaskan bahwa: "perancangan adalah kemampuan untuk membuat beberapa alternatif pemecahan masalah.

Menurut Azhar Susanto (2004:332) menjelaskan dalam buku yang berjudul Sistem Informasi Manajemen Konsep dan Pengembangannya yaitu: perancangan adalah spesifikasi umum dan terinci dari pemecahan masalah berbasis komputer yang telah dipilih selama tahap analisis.

Berdasarkan dua definisi perancangan tersebut, maka penulis dapat menyimpulkan bahwa perancangan merupakan suatu alternatif untuk memecahkan masalah dan yang telah dipilih selama tahap analisis dalam pemecahan masalah yang dihadapi perusahaan.

II.2. Sistem

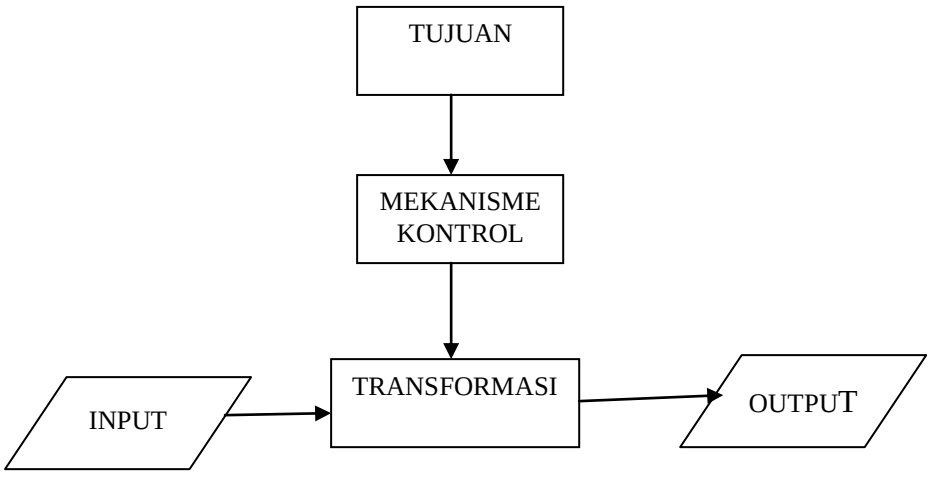
Menurut Hall (2001:5), sistem adalah sekelompok dua atau lebih komponen - komponen yang saling berkaitan (*interrelated*) atau sub elemen – sub elemen yang bersatu untuk mencapai tujuan yang sama (*common purpose*).

Menurut Tata Sutabri (2003:18), Sistem adalah suatu jaringan prosedur yang di buat menurut pola yang terpadu untuk melaksanakan kegiatan pokok perusahaan.

Menurut James A Hall (2007:6), sistem adalah kelompok dari dua atau lebih komponen atau subsistem yang saling berhubungan yang berfungsi dengan tujuan yang sama.

Menurut Murdick dan Ross (2007:1), mendefenisikan sistem sebagai perangkat elemen yang di gabungkan satu dengan yang lainnya untuk suatu tujuan bersama. Sementara, definisi sistem dalam kamus *Webster's unbriged* adalah elemen-elemen yang saling berhubungan dan membentuk satu kesatuan atau organisasi.

Sementara menurut Mc. Leod (2007:2), mendefinisikan sistem sebagai sekelompok elemen-elemen yang berintegrasi dengan maksud yang sama untuk mencapai suatu tujuan. Untuk lebih jelasnya elemen sistem tersebut dapat di gambarkan dengan model sebagai berikut :



Gambar II.1. Model Hubungan Elemem-Elemen Sistem

Sumber : Hanif Al Fatta (2007 : 2)

Dari uraian di atas dapat di ambil kesimpulan bahwa suatu sistem pada dasarnya adalah sekelompok unsur yang erat hubungannya satu dengan yang lain, yang berfungsi bersama-sama untuk mencapai tujuan tertentu (Tata Sutarbi 2005 :8).

II.3. Informasi

Menurut Mc.leod (2001:15), informasi adalah data yang telah diproses, atau data yang yang memiliki arti. Informasi merupakan bagian yang penting dari suatu perusahaan.

Menurut Tata Sutabri (2003:6), Informasi adalah data yang berguna yang di olah sehingga dapat di jadikan dasar untuk mengambil keputusan yang tepat.

Menurut Gordon B. Davis (2005:15), menjelaskan informasi adalah data yang telah di proses ke dalam suatu bentuk yang mempunyai arti bagi si penerima dan mempunyai nilai nyata dan terasa bagi keputusan saat itu dan mendatang.

Sementara menurut Tata Sutabri (2005:23), informasi adalah sebuah istilah yang tidak tepat dalam pemakainya secara umum melainkan informasi adalah data yang telah di *klarifikasi* atau di olah atau *interpretasi* untuk di gunakan dalam proses pengambilan keputusan. Sistem pengolahan informasi mengolah data menjadi informasi atau tepatnya mengolah data dari bentuk tak berguna menjadi berguna bagi penerimanya.

II.4. Sistem Informasi

Menurut James A. Hall (2007:9), sistem informasi *(Information system)* adalah serangkaian prosedur formal dimana data dikumpulkan, di proses menjadi informasi dan didistribusikan ke para pengguna.

Menurut Burch dan Grudnistki (2007:10), sistem informasi yang terdiri dari komponen-komponen diatas disebut sebagai istilah blok bangunan (*building block),* yaitu blok masukan (*input block),* blok model (*model block),* blok keluaran (*out put block),* blok teknologi (*tecnologi block),* dan blok kendali (*control block).* Sebagai suatu sistem , keenam blok tersebut masing-masing saling berinteraksi satu dengan yang lainnya membentuk satu kesatuan untuk mencapai sasarannya.

Menurut Tata sutabri (2005:42), sistem Informasi adalah suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian yang mendukung sistem operasi organisasi yang bersifat manajerial dengan kegiatan strategi dari suatu organisasi untuk dapat menyediakan kepada pihak luar tertentu dengan laporan-laporan yang di tentukan.

Berdasarkan pengertian diatas Sistem Informasi yang dikemukakan oleh para ahli dapat disimpulkan bahwa pengertian sistem informasi adalah satuan komponen yang saling berkaitan antara satu dan lainnya melalui proses, dimana proses tersebut menghasilkan data yang memiliki nilai Komponen Sistem informasi Sebagimana yang telah diuraikan sebelumnya bahwa sistem terdiri dari kumpulan komponen atau sub sistem, sistem informasi juga terdiri dari komponen-komponen yang saling berintegrasi satu dengan yang lainnya membentuk satu kesatuan dalam mencapai sasaran.

II.5. Data

Menurut Gordon, B. Davis (2005 : 16) Data merupakan bentuk jamak dari bentuk tunggal *datum.* Data adalah kenyataan yang menggambarkan suatu kejadian-kejadian dan kesatuan nyata. Kejadian-kejadian adalah suatu yang terjadi pada saat tertentu di dalam dunia bisnis adalah perubahan dari suatu nilai yang di sebut transaksi. Dari definisi dan uraian data tersebut dapat di simpulkan bahwa data adalah bahan mentah yang di proses untuk menyajikan informasi .

Menurut Drs. John J.Longkutoy dalam bukunya "pengenalan komputer" sebagai berikut : istilah data adalah suatu istilah majemuk yang berarti fakta atau bagian dari fakta yang menganung arti yang di hubungkan dengan kenyataan, simbol-simbol, gambar-gambar, angka-angka, huruf-huruf, atau simbol yang menunjukkan suatu ide, objek, kondisi atau situasi dan lain-lainya. Jelasnya data itu dapat berupa apa saja dan dapat di temui dimana saja . kemudian kegunaan data adalah sebagai bahan dasar yang objektif (relatif) didalam proses penyusunan dan kebijakan keputusan oleh pimpinan organisasi.

II.6. Akuntansi

Dalam buku akuntansi suatu pengantar *American Accounting association* mendefinisikan akuntansi sebagai proses mengidentifikasi, mengukur dan melaporkan informasi ekonomi untuk memungkinkan adanya penilain dan keputusan yang jelas dan tegas bagi mereka yang menggunakan informasi tersebut. Definisi ini mengandung dua pengertian, yakni :

a. Kegiatan akuntansi

   Bahwa akuntansi merupakan prose yang terdiri dari identifikasi, pengukuran dan pelaporan informasi ekonomi

b. Kegunaan akuntansi

   Bahwa informasi ekonomi yang di hasilkan oleh akuntansi di harapkan berguna dalam penilaian dan pengambilan keputusan mengenai usaha yang bersangkutan.

Definisi akuntansi dalam buku yang berjudul *Praktikum Akuntansi Manual dan Komputerisasi dengan MYOB* karangan Erly Suandy dan Jessica (2008:3), bahwa: "akuntansi dapat didefinisikan sebagai sistem informasi yang menghasilkan laporan mengenai aktivitas ekonomi dan kondisi dari suatu entitas/ perusahaan kepada pihak- pihak yang berkepentingan."

Menurut Soemarso (2009:14), dalam buku yang berjudul *Akuntansi Suatu Pengantar* yang menerangkan bahwa: "akuntansi (*accounting*) suatu disiplin yang menyediakan informasi penting sehingga memungkinkan adanya pelaksanaan dan penilaian jalannya perusahaan secara efisien.

Menurut Tata Sutabri (2003:3), Akuntansi merupakan teknik yang menggambarkan proses yang menghubungkan sumbar data melalui *"channel"* komunikasi dengan para penerima informasi. Akuntansi memiliki siklus yang disebut *"Accounting cycle"* yang memproses bukti transaksi menjadi bentuk informasi yang kita kenal dengan laporan keuangan yang dapat dipergunakan masyarakat untuk proses pengambilan keputusan.

II.7. Sistem Informasi Akuntansi

Menurut Tata Sutarbi (2004 : 6) Sistem Informasi Akuntansi adalah kumpulan sumber daya, seperti manusia dan peralatan yang di atur untuk

mengubah data menjadi informasi. Informasi ini di komuniksikan kepada beragam pengambilan keputusan . Sistem informasi akuntansi (SIA) pewujudkan perubahan ini apakah secara manual / terkomputerisasi.

Sistem informasi akuntansi merupakan sebuah sistem informasi yang mengubah data transaksi bisnis menjadi informasi keuangan yang berguna bagi pemakainya. Tujuan dari sistem informasi akuntansi adalah :

1. Mendukung operasi sehari-hari
2. Mendukung pengambilan keputusan manajemen
3. Memenuhi kewajiban yang berhubungan dengan pertanggung jawab

Komponen-komponen yang ada dalam sistem informasi akuntansi adalah :

1. Orang-orang yang mengoperasikan sistem tersebut
2. Prosedur-prosedur, baik manual maupun yang terotomatisasi, yang di libatkan dalam pengumpulan pemrosesan dan penyimpanan aktifitas-aktifitas organisasi.
3. Data tentang proses-proses bisnis
4. Software yang di pakai untuk memproses data organisasi
5. Infrastuktur teknologi informasi.

Pemakai informasi akuntansi dapat di bagi dalam dua kelompok besar, yaitu ekstern dan intern. Permakai ekstern mncakup pemegang saham, investor, kreditor, pemerintah, pelanggan, pemasok, pesaing, serikat pekerja dan masyarakat secara keseluruhan. Pemakai ekstern menerima dan tergantung pada beragam keluaran dari sistem informasi akuntansi suatu organisasi. Dan pemakai

intern terutama para manajer, kebutuhan bervariasi tergantung pada tingkatnya di dalam organisasi.

II.8. Pengukuran Laba

Menurut Donald E. Kieso, Jerry J. Weygandt dan Terry D. Warfield yang diterjemahkan oleh Emil Salim (2002:40) menyatakan bahwa: "Keuntungan adalah kenaikan ekuitas (aktiva bersih) sebuah perusahaan yang ditimbulkan oleh transaksi *peripheral* atau *insedentil* dan dari semua transaksi serta kejadian lainnya dan situasi yang mempengaruhi perusahaan selama suatu periode kecuali yang berasal dari pendapatan atau investasi oleh pemilik.

Menurut Soemarso (2004:132), kemajuan dan pertumbuhan perusahaan dapat di ukur dari kemampuan perusahaan menghasilkan laba. Kemampuan ini tentu saja tidak di ukur dari bentuk laba *absolute* (jumlah laba) yang di peroleh. Akan tetapi, harus di perbandingkan misalnya dengan jumlah modal yang di tanam , jumlah aktiva yang di pakai, jumlah penjualan dan lain-lain. Angka terakhir dalam laporan laba rugi adalah laba bersih (*net profit*). jumlah ini merupakan kenaikan bersih terhadap modal. Sebaliknya, apabila perusahaan menderita rugi, angka terkhir dalam laporan laba rugi adalah rugi bersih (*net loss).* Dari pernyataan diatas ini dapat diketahui bahwa penghasilan dapat dibagi dua yaitu:

1. Laba yang sudah direalisasi yaitu laba yang terjadi dari transaksi penjualan

2. Laba yang belum direalisasi yaitu laba yang terjadi karena pertambahan kekayaan sebagai akibat kenaikan nilai aktiva dan belum terjadi transaksi.

Sedangkan menurut Standar Akuntansi Keuangan (2002:25) pengertian laba terdiri dari dua konsep yaitu:

1. Pemeliharaan modal keuangan, dimana laba hanya diperoleh kalau jumlah *financial* (uang) dari aktiva bersih pada akhir periode melebihi *financial* atau uang dari aktiva bersih pada awal periode, setelah *konstibusi* dari perusahan memiliki selama satu periode.
2. Pemeliharaan modal fisik, dimana laba hanya diperoleh kalau kapasitas produtif fisik (kemampuan usaha) pada awal periode, setelah *konstribusi* dari para pemilik selama satu periode. Berikut ini komponen unsur-unsur laba yang terdiri dari :
   a. Pendapatan yaitu arus masuk atau penambahan lain atas aktiva suatu entitas atau penyelesaian kewajiban – kewajibannya (kombinasi keduanya) yang berasal dari penyerahan atau produksi barang, operasi utama atau operasi inti yang berkelanjutan dari suatu entitas.
   b. Beban yaitu arus keluar atau pemakaian lain aktiva atau terjadinya kewajiban (kombinasi keduanya) yang berasal dari penyerahan atau produksi barang, pemberian jasa, atau pelaksanaan jasa atau pelaksanaan aktivitas-aktivitas lain yang merupakan operasi utama atau operasi ini berkelanjutan dari suatu entitas

c. Keuntungan yaitu kenaikan suatu entitas atau aktivitas besih yang berasal dari transaksi *peripheral* atau *insidental* pada suatu entitas dan dari transaksi lain dan kejadian serta situasi lain yang mempengaruhi entitas kecuali yang dihasilkan dari pendapatan atau investasi pemilik.

II.8.1. Metode Langsung

Menurut Johan Arifin (20:2008) Bentuk *Single Step* atau Langsung di susun dengan mengelompokkan Semua pendapatan atau penghasilan kedalam satu kelompok yang di sebut penghasilan. Semua beban yang terjadi dalam satu periode dalam satu kelompok yang di sebut biaya. Selisih positif antara kelompok penghasilan dengan biaya di sebut istilah penghasilan bersih atau laba, sedangkan jika selisih negative di sebut rugi. Laporan laba rugi yang menggunakan langkah langsung tidak membedakan beban penjualan dengan beban umum dan dministrasi. Metode ini banyak di gunakan oleh perusahaan kecil dengan transaksi yang tidak terlalu banyak.

Langkah-langkah Penyusunan Laporan pengukuran Laba dengan metode Langsung. Beberapa hal yang harus diperhatikan dalam menyusun Laporan Pengukuran Laba, yaitu :

1. Judul Laporan

   Menuliskan nama perusahaan, nama laporan, dan periode laporan di tengah atas halaman.

2. Isi Laporan bentuk *single step (*Langsung) *:*

a. Menuliskan semua pendapatan

b. Menuliskan semua beban

c. Menghitung selisih pandapatan dan beban, jika pendapatan lebih besar dari pada beban maka selisihnya disebut laba bersih dan jika sebaliknya maka selisihnya disebut rugi bersih.

Contoh bentuk Laporan Pengukuran Laba dengan menggunakan metode langsung (*Singel Step*)

CV.CITRA MANDIRI MEDAN
LAPORAN LABA-RUGI
PER 31 DESEMBER 1999

Pendapatan Usaha		
1. Pendapatan Penjualan		Rp.100.000.000,00
2. Pendapatan Bunga		Rp. 1.200.000,00
JumlahPendapatan		Rp.101.200.000,00
Beban Usaha		
1.Beban Gaji	Rp.2.000.000,00	
2.Beban Penyusutan Peralatan	Rp. 800.000,00	
3.Beban Asuransi	Rp. 50.000,00	
4.Beban Perlengkapan	Rp. 400.000,00	
5.Beban Bunga	Rp. 400.000,00	
Jumlah Beban Usaha		Rp. 3.650.000,00
Laba Bersih		Rp.97.350.000,00

Gambar II.2. Contoh Pengukuran Laba dengan Metode Langsung

II.9. Konsep Dasar Bahasa Pemrograman Visual Studio. Net

II.9.1. Sekilas Tentang Visual Studio. Net

Menurut Rahmat Priyanto (2009:1), *visual basic* 2008 merupakan salah satu paket bahasa pemograman dari *visual studio* 2008. Banyak fasilitas yang

akan kita dapatkan melalui rilis *visual basic* versi ini. Visual studio 2008 sendiri merupakan sebuah *software* untuk membuat aplikasi *windows*, jadi melalui *software* ini kita bisa membuat sebuah aplikasi *data base*, aplikasi *inventory* dan sebagainya. *Visual Studio.Net* adalah sebuah *tool* pengembangan perangkat lunak untuk membangun *aplikasi ASP Web*, layanan *XML Web*, aplikasi dekstop dan *aplikasi mobile*. *Visual Studio.Net , Visual C++. Net , Visual C# .Net* dan *Visual J#.Net*, semuanya menggunakan i*ntegrated devolopment enviroment (IDE).* Atau lingkungan pengembangan *terintegrasi* yang sama yang membolehkan mereka untuk saling berbagi *tools* dan fasilitas dalam pembuatan solusi yang memadukan beberapa bahasa *(mixed longuage solution).*

II.9.2. Lingkungan Pemrograman *Visual Studio. Net*

Layar *visual studio* 2008 sama dengan layar program-program *aplikasi windows* pada umumnya, kita dapat memindah-mindahkan, menggeser, memperbesar, atau memperkecil ukuran setiap komponen layar *visual basic* seperti kita memanipulasi jendela *windows*.

Komponen-komponen dari lingkungan visual studio 2008 dapat dilihat pada Gambar II.3. (Rahmat Priyanto 2009:2).

Gambar II.3. Tampilan utama Visual Studio 2008

Sumber : Rahmat Priyanto (2009:2)

Melalui gambar kerja ini kita dapat membuat aplikasi visual basic yang di perlukan. Sebelum kita membuat aplikasi baru, ada dua buah istilah yang perlu kita ketahui dalam visual studio 2008 yaitu :

1. Project, merukan sebutan bagi sebuah software yang sedang malalui tahap pembuatan menggunakan visual studio, belum menjadi sebuah aplikasi. Terdapat berbagai jenis project pembuatan aplikasi windows, project pembuatan aplikasi console, dan sebagainya.
2. Solution, adalah kumpulan beberapa buah project, sebuah solution dapat terdiri atas satu buah project atau beberapa buah project, bergantung pada kebutuhan. Sebuah project harus di simpan dalam sebuah solution.

II.10. *Microsoft SQL Server*

SQL Server adalah salah satu produk *Relational Database Management Sistem* (RDBMS) populer saat ini. Fungsi utamanya adalah sebagai *database server* yang mengatur semua proses penyimpanan data dan transaksi suatu aplikasi. *SQL Server Versi 2008* memiliki *feature-feature* lengkap untuk membangun aplikasi mulai skala kecil sampai dengan tingkat *enterprise*. Untuk lebih jelasnya dapat dilihat pada Gambar II.2 (Wahana Komputer ; 2008 : 40)

Gambar II.4. *SQL Server 2008*

Sumber : Wahana Komputer (2008 :40)

II.10.1. *Interface SQL Server 2008*

Ada 3 *interface* utama saat bekerja dengan *SQL Server* 2008 adalah sebagai berikut :

1. *Registered Server*

Bila pada tampilan pertama anda tidak melihat panel ini maka anda dapat menampilkannya pada menu *View Regristed Servers* atau menekan kombinasi tombol CTRL + ALT + G. Panel ini memungkinkan anda menjaga koneksi-koneksi dengan server-server yang pernah digunakan. Koneksi-koneksi ini dapat digunakan untuk memeriksa dari server tersebut (*online* atau *offline*) atau melakukan pada obyek-obyeknya (menggunakan pada panel *object explorer*). Setiap user memiliki daftar tersendiri dari *regristered server* yang disimpan pada mesin lokal. Anda dapat melakukan penambahan atau pengurangan koneksi ke *server*. Anda dapat mengkelompokan koneksi – koneksi ke server tersebut berdasarkan tipe servernya yaitu *Database Engine, Analysis Service, Reporting Service, Intergration Service*.

2. *Object Explorer*

Kita dapat melihat berbagai obyek yang ada pada sebuah server pada panel ini. Bila panel ini tidak terlihat maka anda dapat menampilkannya dengan menu *View Object Explorer*. Apabila anda melakukan ekpansi dari sebuah cabang maka sebuah struktur logika dari sebuah obyek akan muncul. Anda dapat mengklik tanda + pada sebelah kiri untuk melakukan ekspansi cabang. Untuk melakukan koneksi pada sebuah server klik kanan pada nama servernya kemudian pilih *Connect*, untuk melakukan *Disconnect* , klik kanan dan pilih *disconnect*

3. *Query Editor*

Jendela ini digunakan untuk membuat dan melakukan *editing* perintah perintah T-SQL dan *mengeksekusi* perintah tersebut. Jendela ini akan muncul otomatis setiap kali anda melakukan kegiatan yang berhubungan dengan *query*. Apabila anda berminat membuat *query* baru atau melakukan *editing file query* yang telah ada, maka jendela ini otomatis akan muncul. Anda dapat membuat *File Query With Current Connection* atau *Database Engine Query atau SQL Server Commpact Query* atau memanfaatkan tombol *New Query* pada *toolbar* (Wahana Komputer ; 2008 : 45-49).

II.10.2. Komponen –Komponen *SQL Server* 2008

Ada 5 komponen-komponen *SQL Server* 2008 adalah sebagai berikut :

1. *Literal Value*

Yang termasuk dengan *literal value* adalah huruf (a – z), *numerik* (0-9), dan hexadesimal (0x). *Literal value* juga dikenal dengan konstanta. Sebuah *konstanta string* terdiri atas satu atau beberapa karakter yang diapit tanda kutip tunggal *(apostroph)* atau tanda petik ganda. Defenisi *konstanta string* sebaiknya digunakan dengan tanda petik tunggal karena tanda petik ganda dalam *query* memiliki fungsi lain (Wahana Komputer 2008: 84).

2. Delimeter

Bahwa sebaiknya menggunakan tanda petik tunggal untuk mengawali dan mengakhiri sebuah konstanta *string* dari pada tanda petik ganda. Hal ini karena tanda petik ganda juga digunakan sebagai delimeter atau pemisah. Tanda kutip ganda tidak dapat digunakan sebagai pembuka dan penutup dari sebuah konstanta

string perintah berikut ini diberikan yaitu *Set Queted Identifier On* Standar perinth tersebut adalah *ON*. Perintah tersebut mengakibatkan tanda petik ganda diatur menjadi *Delimeted Identifier*. *Delimeted Identifier* adalah *identifier* khusus yang memungkinkan reserver word digunakan menjadi *identifier* dan juga membolehkan adanya spasi pada nama *database.*

3. Komentar

Komentar dalam pemograman ataupun *scripting* diperlukan untuk memberikan keterangan singkat tentang kode-kode yang ada dibawahnya. Sehingga sewaktu ada kerusakan, kesalahan, *programmer* dapat dengan mudah menngerti apa kegunaan dari kode tersebut. Pada *SQL Server* terdapat dua macam komentar yaitu :

a. Komentar yang menggunakan tanda / * * /. Dengan tanda ini komentar yang anda berikan dapat terdiri atas beberapa baris di awali dengan / * dan di akhiri dengan */.
b. Komentar yang menggunakan tanda – untuk memberi komentar pada baris yang dimaskud saja. Biasanya digunakan untuk menerangkan *identifier* atau *reserved word*.

4. *Identifier*

Dalam pemograman bahasa *T-SQL* untuk *query*, *identifier* digunakan untuk melakukan identifikasi *database* dan obyek-obyeknya seperti tabel dan *index*. Diidentifikasi dengan string karakter dengan panjang maksimal 128 karakter. *Identifier* ini dapat terdiri atas berbagai macam huruf, angka dan karakter khusus

(@, _#, $). Setiap identifier harus dimulai dengan huruf, atau karakter khusus tidak boleh dengan angka.

5. *Reserved Word*

Reserved word atau kata kunci adalah kata yang memilikiarti khusus dan harus dituliskan dengan aturan tertentu. Dalam bahasa *T-SQL reserved word* dan juga memiliki banyak fungsi. Reserved word tidak dapat digunakan sebagai nama sebuah obyek kecuali obyek tersebut didefenisikan sebagai *delimeted identifier*. (Wahana Komputer ; 2008: 84-86).

II.11. Pengertian Basis Data

Menurut Tata Sutabri (2005:161), *Database* adalah suatu kumpulan data terhubung (interrelated data) yang di simpan secara bersama-sama pada suatu media, tanpa mengatap satu sama lain atau tidak perlu suatu kerangkapan data *(contolled redundancy)* dengan cara tertentu sehingga mudah di gunakan atau di tampilkan kembali, dapat di gunakan satu atau lebih program aplikasi secara optimal . data di simpan tanpa mengalami ketergantungan pada program yang akan menggunakannya. Data di simpan sedemikian rupa sehingga penambahan, pengambilan dab modifikasi dapat di lakukan dengan mudah dan terkontrol.

Data base merupakan salah satu komponen yang penting dalam menyediakan informasi bagi para pemakai.Penerapan *Data Base* dalam sistem informasi disebut dengan *Data Base System*, yaitu suatu sistem informasi yang mengintegrasikan kumpulan dari data yang saling berhubungan satu dengan yang lainnya dan

membuatnya tersedia untuk beberapa aplikasi yang bermacam-macam didalam suatu organisasi.

II.11.1. Tingkatan Susunan Organisasi Data

Sebelum mencapai atau membentuk suatu *database*, data mempunyai tingkatan mulai dari karakter-karakter (*characters),* item data (*data item atau field*), *record, file* kemudian *database*. Tingkatan susunan organisasi adalah:

1. Karakter-karakter

Karakter merupakan bagian data yang terkecil, dapat nerupa karakter numerik, huruf ataupun karakter-karakter khusus (*special characters*) yang membentuk suatu item data.

2. *Field*

Field menggambarkan suatu attribut dari record yang menunjukkan suatu item data.

3. *Record*

Record yaitu kumpulan dari *field* yang menggambarkan suatu unit dari data tertentu atau sekumpulan data item yang berhubungan secara logika dari suatu objek.

4. *File*

File merupakan kumpulan dari *record* yang menggambarkan suatu kesatuan yang sejenis.

5 *Database*

Database adalah kumpulan dari *file* yang membentuk satu kesatuan tertentu atau suatu kumpulan data terhubung yang tersimpan secara bersama-sama pada

suatu media, tanpa adanya suatu kerangkapan data sehingga mudah untuk digunakan kembali, dapat digunakan oleh satu atau lebih program aplikasi secara optimal, data disimpan tanpa mengalami ketergantungan pada program yang akan digunakan, data disimpan sedemikian rupa sehingga apabila penambahan, pengambilan dam modifkasi data dapat juga dilakukan dengan mudah dan terkontrol. terdiri dati type data (yang berupa karakter dan *numerik*), *filed, record, file dan database*.

II.10.2. Komponen-komponen Sistem Basis Data

Menurut Tata Sutabri (2005:171) Sistem basis data (*data base*) mempunyai beberapa elemen-elemen penyusun sistem, elemen-elemen pokok penyusun sistem *data base* adalah sebagai berikut :

1. *Database* (basis data) adalah kumpulan *file-file* yang saling berhubungna atau berelasi sehingga membentuk suatu database.
2 *Software* (perangkat lunak) adalah perangkat lunak yang digunakan dalam suatu sistem basisdata (*data base)*. Misalnya: *Foxpro, SQL, Microsoft Access*.
3 *Hardware* (Perangkat Keras), merupakan perangkat keras yang digunakan dalam sistem basis data yaitu unit pusat pengolah (*Central Processing Unit* atau CPU), unit penyimpanan (*Storage Unit*), *Keyboard, Monitor, Printer* dan lai-lain.
4 *Brainware* (manusia), merupakan elemen penting pada sistem basisdata yang terdiri dari sistem *enginer, administrator* basis data, *programmer* dan pemakai terakhir.

II.11. *Entity Relationship Diagram* (ERD)

Merupakan suatu model untuk menjelaskan hubungan antar dua dalam basis data berdasarkan suatu persepsi bahwa *real word* terdiri dari *object-object* dasar yang memmpunyai hubungan atau antar *object-object* tersebut. Relasi antar *object* dengan menggunakan symbol-simbol grafis tertentu.

Model *entity relationship* adalah suatu penyajian dengan menggunakan *entity* dan *relationship*. Diperkenalkan pada tahun 1976 oleh P.P. Chen.

II.12.1. Komponen-komponen yang terdapat didalam *Entity Relationship* Model.

1. *Entity*
    a. Adalah sesuatu yang dapat dibedakan dalam dunia nyata dimana informasi yang berkaitan dengannya dikumpulkan.
    b. *Entity* set adalah kumpulan *entity* yang sejenis.
    c. Symbol yang digunakan untuk *entity* adalah persegi panjang.
    d. *Entity* set dapat berupa :
        1. *Entity* yang bersifat fisik, yaitu *entity* yang dapat dilihat.
           Contohnya : rumah, kendaraan, mahasiswa, dosen, dan lain-lain.
        2. *Entity* yang bersifat konsep atau logic, yaitu *entity* yang tidak dapat dilihat. Contohnya : pekerjaan, perusahaan, rencana, mata kuliah, dan lain-lain.
    e. Simbol yang digunakan untuk *entity* adalah persegi panjang.

    Untuk melihat gambar *entity* ini, lihat pada gambar II.2. sebagai berikut :

Gambar II.5. *Entity*

(Sumber : Linda Marlinda, S. Kom; 2004:17)

2. *Relationship*

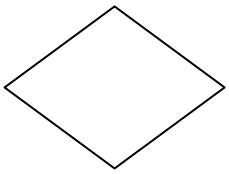
   a. Adalah hubungan yang terjadi antara satu atau lebih *entity*.

   b. *Relationship* tidak mempunyai keberadaan fisik, kecuali yang mewarisi hubungan antara *entity* tersebut.

   c. *Relationship* set adalah kumpulan relationship yang sejenis.

   d. Simbol yang digunakan adalah bentuk belah ketupat, *diamond* atau *rectangle.*

   Untuk melihat gambar *relationship* ini, lihat pada gambar II.3. sebagai berikut :



Gambar II.6. *Relationship*

(Sumber : Linda Marlinda, S. Kom; 2004:18)

3. Attribute

   a. Adalah karakteristik dari *entity* atau *relationship* yang menyediakan penjelasan detail tentang atau *relationship* tersebut.

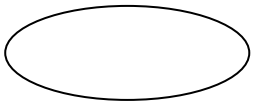
   b. Attribute value (nilai atribute) adalah suatu data aktual atau informasi yang disimpan di suatu atribute di dalam suatu *entity* atau *relationship*.

   c. Terdapat dua jenis atribute, yaitu :

1. *Indetifer (key)*, untuk menentukan suatu *entity* secara unik.
2. *Descriptor (nonkey attribute)*, untuk menentukan karakteristik dari suatu *entity* yang tidak unik.

d. Simbol yang digunakan adalah bentuk oval

Untuk melihat gambar *attribute* ini, lihat pada gambar II.4. sebagai berikut :



Gambar II.7. *Attribute*

(Sumber : Linda Marlinda, S. Kom; 2004:18)

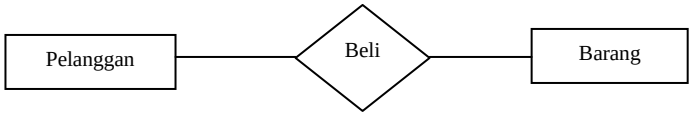
4. Indicator Tipe

a. *Indicator tipe associative object*

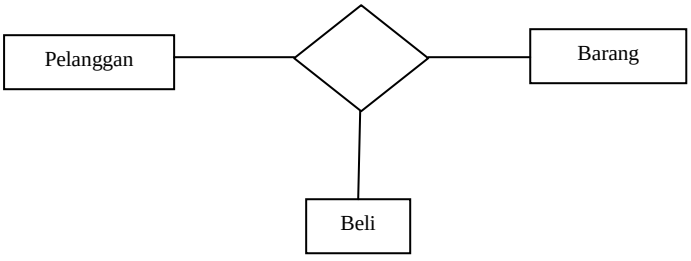
Berfungsi sebagai suatu objek dan suatu *relationship*

Untuk melihat gambar *indicator type* ini, lihat pada gambar II.5. sebagai berikut :

Contoh :



Menjadi :



Gambar II.8. *Indicator Tipe*

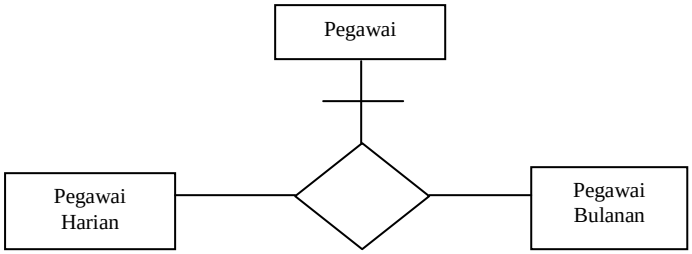
(Sumber : Linda Marlinda, S. Kom; 2004:19)

b. Indicator tipe supertipe

Terdiri dari suatu object dan sub kategori atau lebih yang dihubungkan dengan dihubungkan dengan relationship yang tidak bernama (Linda Marlinda, S. Kom 2004:17-19).

Untuk melihat gambar *indicator tipe supertipe* ini, lihat pada gambar II.6. sebagai berikut :

Contoh :



Gambar II.9. *Indicator Tipe SuperTipe*

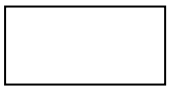
(Sumber : Linda Marlinda, S. Kom; 2004:19)

Keterangan :

Pegawai di kategorikan menjadi 2, yaitu :

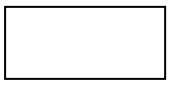
- Pegawai harian
- Pegawai Bulanan

Pegawai mempunyai Nip, Nama, Tahun Masuk, alamat, Nama Supervisor.



Pegawai Harian :

- Upah perjam
- Upah Lembur
- Waktu Mulai kerja



Pegawai Bulanan :

- Gaji perbulan
- Persentasi Bonus Tahunan

II.13. Kamus Data

Kamus data (KD) atau data *dictionary* (DD) yang disebut juga dengan *system dictionary* data adalah katalog fakta tentang data dan kebutuhan-kebutuhan informasi dari suatu sistem informasi. Dengan demikian KD, analisis sistem dapat mendefenisikan data yang mengalir di sistem dengan lengkap. KD dibuat pada pada tahap analisis sistem dan digunakan baik pada tahap analisis maupun pada tahap perancangan sistem. Pada tahap analisis, KD dapat digunakan sebagai alat komunikasi antara analisis sistem dengan pemakai sistem tentang data yang mengalir di sistem. Pada tahap perancangan sistem KD digunakan untuk merancang input, merancang laporan-laporan dan database. KD dibuat berdasarkan arus data yang ada di DAD. Arus data di DAD sifatnya global, hanya ditunjukkan nama arus datanya saja (Prof.Dr.Jogiyanto HM, MBA,Ph.D 2005:725).

II.13.1. Isi Kamus Data

Kamus Data (KD) harus dapat mencerminkan keterangan yang jelas tentang data yang dicatatnya. Untuk maksud keperluan ini maka KD harus memuat hal-hal berikut ini :

1. Nama Arus Data

   Karena KD dibuat berdasarkan arus data yang mengalir di DAD, maka nama dari arus data juga harus dicatat di KD, sehingga mereka yang membaca DAD dan memerlukan penjelasan lebih lanjut tentang suatu arus data tertentu di DAD dapat langsung mencarinya dengan mudah di KD.

2. Alias

   Alias atau nama lain dari data dapat dituliskan bila nama lain ini ada. Alias perlu ditulis karena data yang mempunyai nama yang berbeda untuk orang atau departemen satu dengan yang lainnya. Misalnya bagian pembuat faktur dan langganan menyebut bukti penjualan sebagai faktur, sedang bagian gudang menyebutnya sebagai tembusan permintaan persediaan barang. Baik faktur dan tembusan permintaan persediaan ini mempunyai struktur data yang sama tetapi mempunyai struktur yang berbeda.

3. Bentuk data

   Telah diketahui bahwa arus data 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 biasanya terdapat di media laporan atau *query* tampilan layar atau dokumen hasil cetakan komputer.
   c. Hasil dari suatu proses yang lain, data yang mengalir biasanya dalam bentuk variabel atau parameter yang dibutuhkan oleh proses penerimanya.
   d. Hasil dari 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 :

1. Dokumen dasar atau formulir
2. Dokumen hasi cetakan komputer
3. Laporan tercetak.
4. Tampilan dilayar monitor.
5. Variabel.
6. Parameter
7. *Field*

Bentuk dari data ini perlu dicatat di KD, karena dapat digunakan untuk mengkelompokan KD ke dalam kegunaanya, sewaktu perancangan sistem KD yang mencatat data yang mengalir dalam bentuk dokumen dasar atau formulir yang digunakan untuk merancang bentuk input sistem. KD yang mencatat data yang mengalir dalam bentuk laporan tercetak dan dokumen hasil cetakan komputer akan digunakan untuk merancang *output* yang akan dihasilkan oleh sistem. KD yang mencatat data yang mengalir dalam bentuk tampilan layar monitor akan digunakan juga untuk merancang tampilan layar yang akan dihasilkan oleh sistem. KD yang mencatat data yang mengalir ke dalam bentuk parameter dan variabel akan digunakan untuk merancang proses dari program. KD yang mencatat data yang mengalir ke dalam bentuk dokumen, formulir, laporan, dokumen cetakan komputer, tampilan di layar monitor, variabel, dan *field* akan digunakan untuk merancang database.

4. Arus data

   Arus data menunjukkan dari mana data yang mengalir dan kemana data akan menuju. Keterangan arus data ini perlu dicatat di KD supaya memudahkan mecari arus data di DAD.

5. Penjelasan

   Untuk lebih memperjelas lagi tentang makna dari arus data yang di catat di KD, maka bagian penjelasan dapat diisidengan keterangan-keterangan tentang arus data tersebut. Sebagai misalnya nama dari arus data adalah Tembusan Permintaan Persediaan, maka dapat lebih di jelaskan dari faktur penjualan meminta barang dari gudang.

6. Periode

   Periode ini menunjukkan kapan terjadinya arus data ini. Periode ini perlu di catat di KD karena dapat di gunakan untuk mengindentifikasi kapan input data harus di laporkan ke sistem, kapan proses dari program harus di lakukan dan kapan laporan-laporan haru di hasilkan.

7. Volume

   Volume yang perlu di catat di KD 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 di gunakan untuk mengidentifikasi besarnya simpanan luar yang akan di gunakan, kapasitas dan jumlah dari alat pemrosesan dan alat out put.

8. Stuktur Data

   Stuktur data menunjukkan arus data yang di catat di KD terdiri dari item-item data apa saja (Prof.Dr.Jogiyanto HM, MBA,Ph.D 2005:725-728).

II.14. Normalisasi *(Normalizing the Relation)*

Normalisasi adalah proses pengkelompokkan attribute-attribute dan suatu relasi sehingga membentuk *Well- Structure Relation.* Normalisasi merupakan proses pengkelompokkan elemen data menjadi suatu tabel-tabel menunjukan entity dan relasinya. Normalisasi ditemukan pada tahun 1970 oleh E. F. CODD.

II.14.1. Well-Structure Relation

Well- Structure Relation adalah sebuah *relation* dengan jumlah kerangkapan datanya sedikit (*Minimum amount of redundancy*), serta memberikan kemungkinan bagi user untuk melakukan *Insert, Delete,* dan *Modify* terhadap baris-baris data pada *relation* tersebut, yang tidak berakibat terjadinya *Error* atau INKONSENTENSI DATA, yang disebabkan oleh operasi-operasi tersebut.

Contoh :

Terdapat sebuah *Relation Course* dengan ketentuan sebagai berikut :

a. Setiap mahasiswa hanya boleh mengambil satu mata kuliah saja.
b. Setiap mata kuliah mempunyai uang kuliah yang standar (tidak tergantung pada mahasiswa yang mengambil mata kuliah tersebut).

Tabel II.1. *Relation Course*

STUDENT-ID	KODE-MTK	BIAYA
92130	CS-200	75
92200	CS-300	100
99250	CS-300	75
92425	CS-400	150
92500	CS-300	100
92575	CS-500	50

(Sumber : Linda Marlinda, S. Kom; 2004 :115)

Relation Course diatas merupakan sebuah *relation* yang sederhana dan terdiri dari 3 kolom/attribute (Linda Marlinda; 2004 : 115).

Ada beberapa Bentuk Normalisasi, Yaitu :

a. Bentuk tidak normal (*Unnormalized Form*)

Bentuk ini merupakan kumpulan data yang akan direkam, tidak ada keharusan mengikuti suatu format tertentu. Dapat saja data tidak lengkap atau terduplikasi. Data dikumpulkan apa adanya dengan saat menginput.

Contoh data :

Tabel II.2. Bentuk tidak normal (*Unnormalized Form*)

No_Siswa	Nama	Pa	Kelas 1	Kelas 2	Kelas 3
22890100	Shandy	Linda	1234	1543	1543
22890101	Susi	Riska	1234	1775	

(Sumber : Linda Marlinda, S. Kom; 2004 : 122)

Siswa yang punya nomor siswa, nama, dan pa mengikuti 3 mata pelajaran/kelas. Di sini ada perulangan kelas 3 kali ini bukan bentuk tidak 1 NF.

b. Bentuk Normal Ke Satu (1 NF/ *Fisrt Normal Form*)

Suatu relasi 1 NF dan hanya jika sifat dan setiap relasi attributenya bersifat *atomic*. Atom adalah zat terkecil yang masih memiliki sifat induknya. Bila dipecah lagi maka ia tidak memiliki sifat induknya.

Ciri-ciri 1 NF :

1. Setiap data dibentuk kedalam *flat file* data terbentuk per satu *record* nilai dan field berupa "*atomic value*".
2. Tidak ada *set attribute* yang berulang atau bernilai ganda,
3. Tiap *field*
4. hanya satu pengertian.

Tabel II.3. Bentuk Normal Ke Satu (1 NF/ *Fisrt Normal Form*)

No_Siswa	Nama	Pa	Kelas 1
22890100	Shandy	Linda	1234
22890100	Shandy	Linda	1543
22890101	Susi	Riska	1234
22890101	Susi	Riska	1775
22890101	Susi	Riska	1543

(Sumber : Linda Marlinda, S. Kom; 2004 : 122)

c. Bentuk Normal Ke Dua (2 NF/ *Second Normal Form*)

Bentuk normal kedua mempunyai syarat yaitu bentuk data telah memenuhi kriteria bentuk normal kesatu. *Attribute* bukan kunci haruslah bergantung secara fungsi pada *primary key*. Jadi, untuk membentuk normal kedua haruslah sudah ditentukan kunci-kunci *field.* Kunci *field* haruslah unik dan dapat mewakili *attribute* lain yang menjadi anggotanya. Misal : dari contoh relasi siswa pada 1 NF terlihat bahwa *primary key* adalah nomor siswa. Nama siswa dan pa bergantung fungsi pada no_siswa, tetapi kode_kelas bukanlah fungsi dan siswa dipecah menjadi 2 relasi :

Relasi siswa :

Tabel II.4. Bentuk Normal Kedua Relasi Siswa (2 NF/ *Second Normal Form*)

No_Siswa	Nama	Pa
22890100	Shandy	Linda
22890101	Susi	Riska

(Sumber : Linda Marlinda, S. Kom; 2004 : 123)

Relasi ambil_Kelas

Tabel II.5. Bentuk Normal Kedua Relasi ambil_Kelas (2 NF/ *Second Normal Form)*

No_Siswa	Kode Kelas
22890100	1234
22890100	1543
22890101	1234
22890101	1775
22890101	1543

(Sumber : Linda Marlinda, S. Kom; 2004 : 123)

d. Bentuk Normal Ketiga (3 NF/*Third Normal Form*)

Untuk menjadi bentuk normal ketiga maka relasi dalam bentuk normal kedua dan semua *attribute* bukan primer tidak punya hubungan yang transistif. Dengan kata lain, setiap *attribute* bukan kunci haruslah bergantung hanya pada *primary key* dan *primary key* secara menyeluruh.

Contoh pada bentuk normal kedua diatas termasuk juga bentuk normal ketiga karena seluruh attribute yang ada bergantung penuh pada kunci primernya.

e. *Boyee-Cood Normal Form* (BCNF)

BNCF mempunyai paksaan lebih kuat dan bentuk normal ketiga untuk menjadi BCNF, relasi harus dalam bentuk normal kesatu dan setiap *attribute* harus bergantung fungsi pada *attribute superkey.*

Pada contoh di bawah ini terdapat relasi seminar dengan ketentuan sebagai berikut :

1. Kunci primer adalah no_siswa +seminar
2. Siswa bopleh mengambil satu atau dua seminar.
3. Setiap siswa dibimbing oleh salah satu di antara 2 instruktur seminar tersebut.
4. Setiap instruktur boleh hanya mengambil satu seminar saja.

Pada contoh ini no_siswa dan seminar menunjuk seorang instruktur :

Relasi seminar

Tabel II.6. *Boyee-Cood Normal Form* (BCNF)

No_Siswa	Seminar	Instruktur
22890100	2281	Si doel
22890101	2281	Pak tile
22890102	2291	Mandra
22890101	2291	Basuki
22890109	2291	Basuki

(Sumber : Linda Marlinda, S. Kom; 2004 : 124)

Bentuk relasi seminar adalah bentuk normal ketiga, tetapi tidak BCNF karena nomor seminar masih tergantung fungsi pada instruktur. Jika setiap instruktur dapat mengajar hanya pada satu seminar saja, maka seminar bergantung fungsi pada satu *attribute* bukan *superkey* seperti disyaratkan oleh BCNF. Maka relasi seminar haruslah dipecah menjadi dua yaitu :

Tabel II.7. *Boyee-Cood Normal Form* (BCNF)

Relasi Pengajar

Instruktur	Seminar
Si doel	2281
Pak tile	2281
Mandra	2291
Basuki	2291

No_Siswa	Instruktur
22890100	Si doel
22890101	Pak tile
22890102	Mandra
22890101	Basuki
22890109	Basuki

(Sumber : Linda Marlinda, S. Kom; 2004 : 124)

f. Bentuk Normal Ke Empat (4NF)

Relasi R adalah bentuk 4 NF jika dan hanya jika relasi tersebut juga termasuk BCNF dan semua ketergantungan *multivaluei* adalah juga ketergantungan fungsional.

g. Bentuk Normal Kelima (5 NF)

Disebut juga PINF (*Projection Join Normal Form*) dan 4 NF dilakukan dengan menghilangakan ketergantungan *join* yang merupakan kunci kandidat (Linda Marlinda, S. Kom ; 2004 : 122-125).

II.15. *Unified Modeling Language* (UML)

Menurut Martin Fowler (2005 : 17) *Unified Modeling Language* (UML) adalah keluarga notasi grafis yang didukung oleh meta-model tunggal, yang membantu pendeskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman beroreantasi objek.

Menurut Prabowo pudjo widodo dan Herlawati (2011:6) *UML* singkatan dari *Unufied Modeling Longuage* yang berarti bahasa standar. Chonoles, 2003 :bab 1 mengatakan sebagai bahasa, berarti *UML* memiliki sintaks dan semantik , ketika kita membuat model menggunakan konsep *UML* ada aturan-aturan yang harus diikuti. Bagaimana elemen pada model-model yang kita buat berhubungan satu dengan yang lainnya harus mengikuti standard yag ada. *UML* bukan hanya sekedar diagram, tetapi juga menceritakan konteksnya. Ketika pelanggan memesan suatu dari sistem, bagaimana transaksinya? Bagaimana sistem mengatasi error yang terjadi? Bagaimana keamanan sistem yang kita buat? Dan sebagainya dapat di jawab dengan *UML*.

UML diaplikasikan untuk maksud tertentu, biasanya antara lain untuk :

1. Merancang Perangkat Lunak
2. Sarana komunikasi antara perangkat lunak dengan proses bisnis
3. Menjabarkan sistem secara rinci untuk analisa dan mencari apa yang diperlukan sistem.
4. Mendokumentasikan sistem yang ada, proses-proses dan organisasinya.

UML telah diaplikasikan dalam investasi perbankan,lembaga kesehatan, departemen pertahanan, sistem terdistribusi, sistem pendukung alat kerja, retail, sales, dan supplier.

Blok pembangunan utama *UML* adalah diagram. Beberapa diagram ada yang rinci (jenis *timing diagram*) dan lainya ada yang bersifat umum (misalnya diagram kelas). Para pengembang sistem berorentasikan objek menggunakan bahasa model untuk menggambarkan, membangun dan mendokumentasikan

sistem yang mereka rancang. *UML* memungkinkan para anggota team untuk bekerja sama dalam mengaplikasikan beragam sistem. Intinya, *UML* merupakan alat komunikasi yang konsisten dalam mensuport para pengembang sistem saat ini. Sebagai perancang sistem mau tidak mau pasti menjumpai *UML*, baik kita sendiri yang membuat sekedar membaca diagram *UML* buatan orang lain (Prabowo Pudji Widodo, Herlawati; 2011 : 6-7).

II.15.1. Diagram-*Diagram UML*

Beberapa literatur menyebutkan bahwa *UML* menyediakan Sembilan jenis diagram, yang lain menyebutkan delapan karena ada beberapa yang digabung, misalnya diagram komunikasi, diagram urutan, dan diagram pewaktuan digabung menjadi diagram interaksi. Namun demikian model-model itu dapat dikelompokan berdasarkan sifatnya yaitu statis atau dinamis. Jenis diagram itu antara lain :

1. Diagram Kelas. Bersifat statis. Diagram ini memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi, serta relasi-relasi diagram. Diagram ini umu dijumpai pada pemodelan sistem berorentasi objek. Meskipun bersifat statis, sering pula diagram kelas memuat kelas-kelas.
2. Diagram Paket (*Package Diagram*) bersifat statis. Diagram ini memperlihatkan kumpulan kelas-kelas merupakan bagian dari diagram komponen.
3. Diagram *Use Case* bersifat statis. Diagram ini memperlihatkan himpunan *use-case* dan aktor-aktor (suatu jenis khusus dari kelas). Diagram ini terutama

sangat penting untuk mengorganisasi dan memodelkan perilaku suatu sistem yang dibutuhkan serta diharapkan pengguna.

4. Diagram Interaksi dan *Sequence* (urutan). Bersifat dinamis. Diagram urutan adalah diagram interaksi yang menekankan pada pengiriman pesan dalam waktu tertentu.
5. Diagram Komunikasi (*Communication Diagram*) bersifat dinamis. Diagram sebagai pengganti diagram kolaborasi *UML* yang menekankan organisasi *structural* dari objek-objek yang menerima serta mengirim pesan.
6. Diagram *Statechart* (*Statechart Diagram)* bersifat dinamis. Diagram status memperlihatkan keadaan-keadaan pada sistem, memuat status (*State),* transisi kejadian serta aktifitas. Diagram ini terutama penting untuk memperlihatkan sifat dinamis dari antarmuka (*interface*), kelas, kolaborasi dan terutam penting pada pemodelan sistem-sistem yang reaktif.
7. Diagram Aktivitas (*Activity Diagram*) bersifat dinamis. Diagram aktivitas adalah tipe khusus dari diagram status yang memperlihatkan aliran dari suatu sistem. Diagram ini terutama penting dalam pemodelan fungsi-fungsi suatu sistem dan member tekanan pada aliran kendali antar objek.
8. Diagram Komponen (*Component Diagram*) bersifat statis. Diagram komponen ini memperlihatkan organisasi serta kebergantungan sistem/perangkat lunak pada komponen-komponen yang telah ada sebelumnya. Diagram ini berhubungan diagram kelas dimana komponen dipetakan kedalam satu atau lebih kelas-kelas. Antarmuka-antarmuka serta kolaborasi-kolaborasi.

9. Diagram *Deployment* (*Deployment Diagram)* bersifat statis. Diagram ini memperlihatkan konfigurasi saat aplikasi dijalankan (*run time)*. Memuat simpul-simpul berserta komponen-komponen yang ada di dalamnya. Diagram *Deployment* berhubungan erat dengan diagram komponen dimana diagram ini memuat satu atau lebih komponen-komponen. Diagram ini sangat berguna saat aplikasi kita berlaku sebagai aplikasi yang dijalankan pada banyak mesin (*distributed computing).*

Kesembilan diagram ini tidak mutlak harus digunakan dalam pengembangan perangkat lunak, semuanya dibuat sesuai dengan kebutuhan. Pada *UML* dimungkinkan kita menggunakan diagram-diagram lainya misalnya *Data Flow Diagram, Entity Relationship Diagram* dan sebagainya (Probowo Pudji Widodo, Herlawati; 2011 : 10-12).

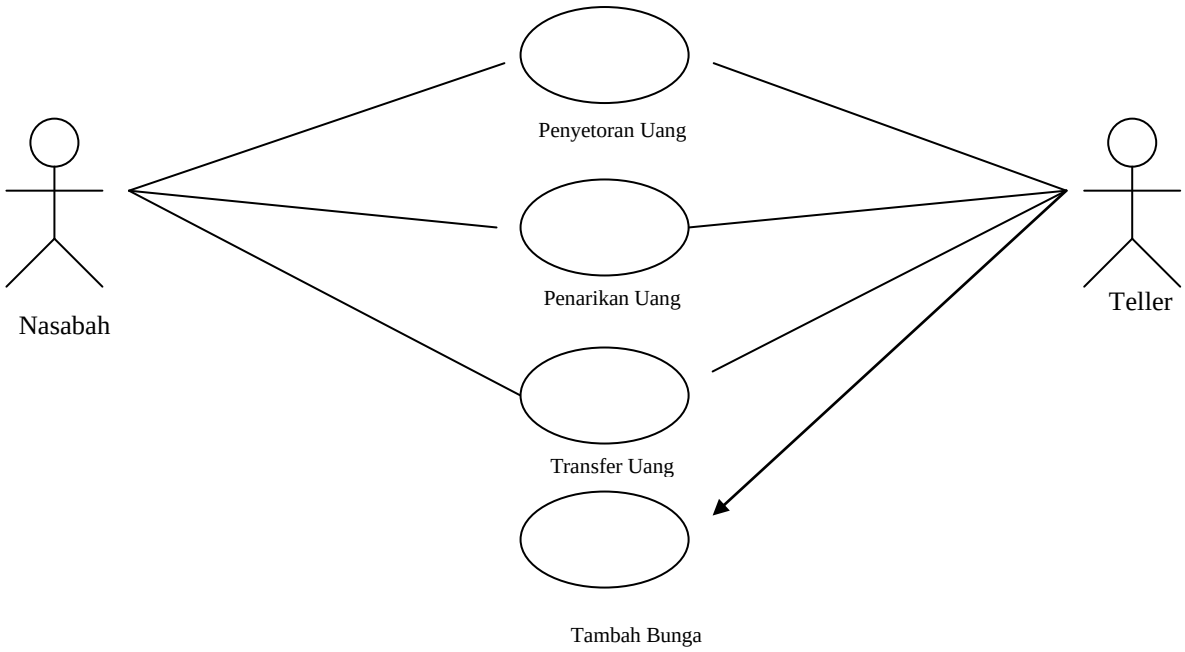
1. *Diagram Use Case (Use Case Diagram)*

   Use Case menggambarkan *external view* dari sistem yang akan kita buat modelnya. Menurut Pooley (2005:15) mengatakan bahwa model *use case* dapat dijabarkan dalam diagram, tetapi yang perlu diingat, diagram tidak indentik dengan model karena model lebih luas dari diagram.

   komponen pembentuk diagram *use case* adalah :

   a. Aktor (*actor),* menggambarkan pihak-pihak yang berperan dalam sistem.
   b. *Use Case*, aktivitas/ sarana yang disiapkan oleh bisnis/sistem.
   c. Hubungan (*Link),* aktor mana saja yang terlibat dalam *use case* ini.

   Gambar II.10. di bawah ini merupakan salah satu contoh bentuk diagram *use case* (Prabowo Pudji Widodo, Herlawati; 2011 : 15-16).

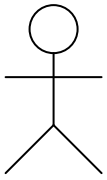


Gambar II.10. Diagram *Use Case*

(Sumber : Prabowo Pudio Widodo, Herlawati; 2011:17)

2. Aktor

   Menurut Chonoles (2003 :17) menyarankan sebelum mebuat use case dan menentukan aktornya, agar mengidentifikasi siapa saja pihak yang terlibat dalam sistem kita. Pihak yang terlibat biasanya dinamakan *stakeholder* untuk melihat gambar aktor, lihat pada gambar II.8. sebagai berikut :

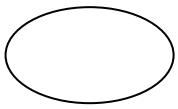


Gambar II.11. Aktor

(Sumber : Prabowo Pudjo Widodo, Herlawati; 2011:17)

3. *Use Case*

Menurut Pilone (2005 : 21) *use case* menggambarkan fungsi tertentu dalam suatu sistem berupa komponen kejadian atau kelas. Sedangkan menurut Whitten (2004 : 258) mengartikan *use case* sebagai urutan langkah-langkah yang secara tindakan saling terkait (skenario) baik terotomatisasi maupun secara manual, untuk tujuan melengkapi satu tugas bisnis tunggal. *Use case* digambarkan dalam bentuk *ellips*/oval untuk melihat gambar simbol *use case*, lihat pada gambar II.12. sebagai berikut :



Gambar II.12. Simbol *Use Case*
(Sumber : Prabowo Pudjo Widodo, Herlawati; 2011:22)

Use case sangat menentukan karakteristik sistem yang kita buat, oleh karena itu Chonoles (2003:22-23) menawarkan cara untuk menghasilkan *use case* yang baik yakni :

a. Pilihlah Nama Yang Baik

Use case adalah sebuah *behaviour* (prilaku), jadi seharusnya dalam frase kata kerja. Untuk membuat namanya lebih detil tambahkan kata benda mengindikasikan dampak aksinya terhadap suatu kelas objek. Oleh karena itu diagram *use case* seharusnya berhubungan dengan diagram kelas.

b. Ilustrasikan Perilaku Dengan Lengkap.

Use case dimulai dari inisiasi oleh aktor primer dan berakhir pada aktor dan menghasilkan tujuan. Jangan membuat *use case* kecuali anda mengetahui

tujuannya. Sebagai contoh memilih tempat tidur (*King Size, Queen Size,* atau dobel) saat tamu memesan tidak dapat dijadikan *use case* karena merupakan bagian dari *use case* pemesanan kamar dan tidak dapat berdiri sendiri (tidak mungkin tamu memesan kamar tidur jenis king tapi tidak memesan kamar hotel).

c. Identifikasi Perilaku Dengan Lengkap.

Untuk mencapai tujuan dan menghasilkan nilai tertentu dari aktor, use case harus lengkap. Ketika memberi nama pada *use case*, pilihlah frasa kata kerja yang implikasinya hingga selesai. Misalnya gunakan frasa *reserve a room* (pemesanan kamar) dan jangan *reserving a room* (memesan kamar) karena memesan menggambarkan perilaku yang belum selesai.

d. Menyediakan *Use Case* Lawan (*Inverse)*

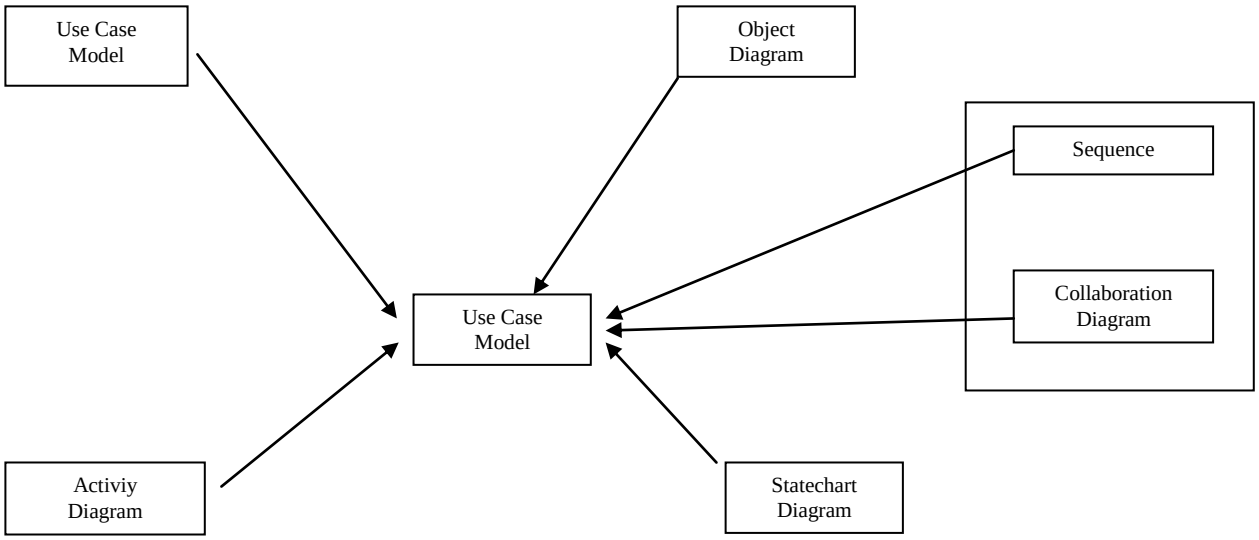
Kita biasanya membutukan *use case* yang membatalkan tujuan, misalnya pada u*se case* pemesanan kamar, dibutuhkan pula *use case* pembatalan pesanan kamar.

e. Batasi Use Case Hingga Satu Perilaku Saja.

Kadang kita cenderung membuat *use case* yang lebih dari satu tujuan aktivitas. Guna menghindari kerancuan, jagalah use case kita hanya fokus pada satu hal. Misalnya, penggunaan *use case check in* dan *check out* dalam satu *use case* menghasilkan ketidakfokusan, karena memiliki dua perilaku yang berbeda (Prabowo Pudji Widodo, Herlawati; 2011 : 22-23).

4. Diagram Kelas (*Class Diagram*)

   Diagram kelas adalah inti dari proses pemodelan objek. Baik *forward engineering* maupun *reverse engineering* memanfaatkan diagram ini *forward engineering* adalah proses perubahan model menjadi kode program sedangkan *reverse engineering* sebaliknya merubah kode program menjadi model (Probowo Pudji Widodo, Herlawati; 2011 : 37) untuk melihat gambar hubungan diagram kelas dengan diagram *uml* lainya, lihat pada gambar II.13. sebagai berikut :



Gambar II.13. Hubungan Diagram Kelas Dengan Diagram *UML* lainya (Sumber : Prabowo Pudjo Widodo, Herlawati; 2011 : 38)

5. Diagram Aktivitas (*Activity Diagram*)

   Diagram aktivitas lebih memfokuskan diri pada eksekusi dan alur sistemdari pada bagaimana sistem dirakit. Diagram ini tidak hanya memodelkan software melainkan memodelkan bisnis juga. Diagram aktivitas menunjukan aktivitas

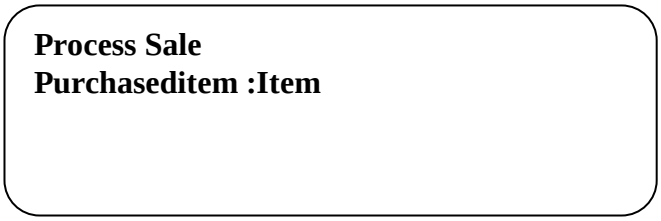
sistem dalam kumpulan aksi-aksi. Ketika digunakan dalam pemodelan *software,* diagram aktivitas merepresentasikan pemanggilan suatu fungsi tertentu misalnya *call.* Sedangkan bila digunakan dalam pemodelan bisnis, diagram ini menggambarkan aktivitas yang dipicu oleh kejadian-kejadian diluar seperti pemesanan atau kejadian-kejadian internal misalnya penggajian tiap jumat sore (Prabowo Pudjo Widodo, Herlawati ;2011 : 143-145)

Aktivitas merupakan kumpulan aksi-aksi. Aksi-aksi nelakukan langka sekali saja tidak boleh dipecah menjadi beberapa langkah-langkah lagi. Contoh aksinya yaitu :

a. Fungsi Matematika
b. Pemanggilan Perilaku
c. Pemrosesan Data

Ketika kita menggunakan diagram aktivitas untuk memodelkan perilaku suatu *classfier* dikatakan kontek dari aktivitas. Aktivitas dapat mengakses atribut dan operasi *classfier,* tiap objek yang terhubung dan parameter-parameter jika aktivitas memiliki hubungan dengan perilaku. Ketika digunakan dengan model proses bisnis, informasi itu biasanya disebut *process-relevant data*. Aktivitas diharapkan dapat digunakan ulang dalam suatu aplikasi, sedangkan aksi biasanya *specific* dan digunakan hanya untuk aktivitas tertentu.

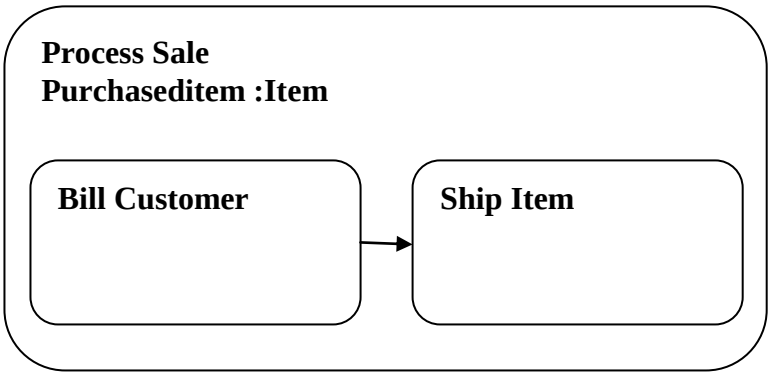
Aktivitas digambarkan dengan persegi panjang tumpul. Namanya ditulis di kiri atas. Parameter yang terlibat dalam aktivitas ditulis dibawahnya. Untuk melihat gambar aktivitas serderhana tanpa rincian, lihat pada gambar II.11. sebagai berikut :



Gambar II.14. Aktivitas Serderhana Tanpa Rincian

(Sumber : Prabowo Pudjo Widodo, Herlawati; 2011:145)

Detail aktivitas dapat dimasukan di dalam kotak. Aksi diperlihatkan dengan symbol yang sama dengan aktivitas dan namanya diletakkan didalam persegi panjang. Untuk melihat gambar aktivitas dengan detail rincian, lihat pada gambar II.15. sebagai berikut :



Gambar II.15. Aktivitas Dengan Detail Rincian

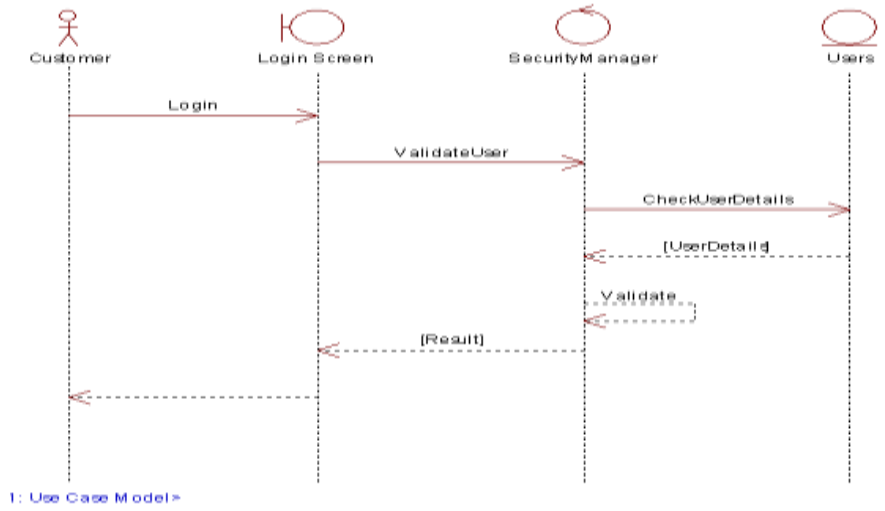
(Sumber : Prabowo Pudjo Widodo, Herlawati; 2011:145)

6. *Sequence Diagram*

Menurut Douglas (2004 : 174) menyebutkan ada tiga diagram primer UML dalam memodelkan scenario interaksi, yaitu diagram urutan (*sequence diagram),* diagram waktu (*timing diagram)* dan diagram komunikasi (*communication diagram).*

Menurut Pilone (2005 : 174) menyatakan bahwa diagram yang paling banyak dipakai adalah diagram urutan. Gambar II.16. memperlihatkan contoh diagram

urutan dengan notasi-notasinya yang akan dijelaskan nantinya (Prabowo Pudji Widodo, Herlawati; 2011 : 174-175)



Gambar II.16. Diagram Urutan

(Sumber : Prabowo Pudjo Widodo, Herlawati; 2011:175)