BAB II

TINJAUAN PUSTAKA

II.1. Sistem

Secara umum sistem dapat di definisikan sebagai sekumpulan objek, ide, berikut saling ketergantungan (inter- relasi) di dalam usaha mencapai suatu tujuan atau dengan kata lain, sistem di sebutkan sebagai kumpulan komponen (subsistem fisik maupun non-fisik/logika) yang saling berhubungan satu sama lainnya dan bekerja sama secara harmonis untuk mencapai suatu tujuan (Eddy Prahasta ; 2009 : 89).

Menurut Jerry FithGerald, sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan dan berkumpul bersama-sama untuk melakukan suatu kegiatan atau menyelesaikan suatu sasaran tertentu.

Menurut Ludwig Von Bartalanfy, sistem merupakan seperangkat unsur yang saling terkait dalam suatu antar relasi di antara unsur - unsur tersebut dengan lingkungan.

Menurut Anatol Raporot, sistem adalah suatu kumpulan kesatuan dan perangkat hubungan satu sama lain.

Menurut L. Ackot, sistem adalah setiap kesatuan secara konseptual atau fisik yang terdiri dari bagian-bagian dalam keadaan saling tergantung satu sama lain. (Asbon Hendra ; 2012 : 157).

II.1.1. Karakteristik Sistem

a. Komponen (Component)

Suatu sistem terdiri dari sejumlah komponen yang salng berinteraksi, bekerjasama membentuk satu kesatuan. Komponen-komponen sistem dapat berupa suatu subsistem atau bagian-bagian dari sistem. Setiap sistem, tidak peduli betapa pun kecilnya, selalu mengandung komponen-komponen atau subsistem-subsistem. Setiap subsistem mempunyai sifat-sifat dari sistem untuk menjalankan suatu fungsi tertentu dan mempengaruhi proses secara keseluruhan. Suatu sistem dapat mempunyai suatu sistem yang lebih besar yang disebut **supra sistem.** Sebagai contoh, suatu perusahaan dapat disebut dengan suatu sistem dan industri yang merupakan sistem yang lebih besar yang dapat disebut dengan supra sistem. Kalau dipandang industri sebagai suatu sistem, maka perusahaan dapat disebut sebagai subsistem. Demikian juga bila perusahaan dipandang sebagai suatu sistem, maka sistem akuntansi adalah subsistemnya.

b. Batas Sistem (Boundary)

Batas sistem merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. Batas sistem ini memungkinkan suatu sistem dipandang sebagai suatu kesatuan, karena dengan batas sistem ini fungsi dan tugas dari subsistem yang satu dengan yang lainnya berbeda tetapi tetap saling berinteraksi. Batas suatu sistem menunjukkan ruang lingkup (scope) dari sistem tersebut.

c. Lingkungan Luar Sistem (Environment)

Environment merupakan segala sesuatu diluar batas sistem yang mempengaruhi operasi dari suatu sistem. Lingkungan luar sistem ini dapat bersifat menguntungkan atau merugikan. Lingkungan luar yang menguntungkan harus dipelihara dan dijaga agar tidak hilang pengaruhnya, sedangkan lingkungan luar yang merugikan harus dimusnahkan atau dikendalikan agar tidak mengganggu suatu sistem.

d. Penghubung Sistem (Interface)

Merupakan media panghubung antara satu sistem dengan subsistem yang lainnya untuk membentuk satu kesatuan sehingga sumber-sumber daya mengalir dari subsistem yang satu ke subsistem yang lainnya. Dengan kata lain, output dari suatu subsistem akan menjadi input dari subsistem yang lain.

e. Masukan Sistem (Input)

Merupakan energi yang dimasukan kedalam sistem. Masukan dapat berupa Masukan Perawatan (Maintenance Input) adalah energi yang dimasukan supaya sistem tersebut dapat beroperasi. Masukan Sinyal (Signal Input) adalah energi yang diproses untuk didapatkan keluaran. Sebagai contoh, didalam sistem komputer, program adalah maintenance input yang digunakan untuk mengoperasikan komputernya dan data adalah signal input untuk diolah menjadi informasi.

f. Keluaran Sistem (Output)

Merupakan hasil dari energi yang diolah oleh sistem, meliputi output yang berguna, contohnya informasi yang dikeluarkan oleh komputer. Dan output yang tidak berguna dikenal sebagai sisa pembuangan, contohnya panas yang dikeluarkan oleh komputer.

g. Pengolah Sistem (Process)

Merupakan bagian dari memproses masukan untuk menjadi keluaran yang diinginkan. Contoh CPU pada komputer, bagian produksi yang mengubah bahan baku menjadi barang jadi, serta bagian akuntansi yang mengolah data transaksi menjadi laporan keuangan.

h. Tujuan Siatem (Goal)

Setiap sistem pasti mempunyai tujuan ataupun sasaran yang mempengaruhi input yang dibutuhkan dan output yang dihasilkan. Dengan kata lain, suatu sistem akan dikatakan berhasil kalau pengoperasian sistem itu mengenai sasaran atau tujuannya. Jika sistem tidak mempunyai sasaran, maka operasi sistem tidak ada gunanya. (Asbon Hendra ; 2011 : 158-160).

II.1.2. Sistem Informasi

Sistem informasi adalah sekumpulan komponen-komponen yang saling berhubungan dan bekerja sama untuk mengumpulkan, memproses, menyimpan, dan mendistribusikan informasi terkait untuk mendukung proses pengambilan keputusan, koordinasi, dan pengendalian. (Eddy Prahasta ; 2009 : 93).

II.1.3. Akuntansi

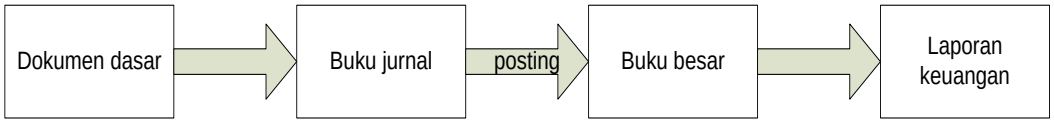
Akuntansi adalah seni daripada pencatatan, penggolongan dan peringkasan daripada peristiwa - peristiwa dan kejadian - kejadian yang setidaknya sebagian bersifat keuangan dengan cara yang tepat dan dengan penunjuk atau dinyatakan dalam uang, serta penafsiran terhadap hal - hal yang timbul daripadanya. (S. Munawir. Akuntan ; 2010 : 05).

Kegiatan selama periode akuntansi adalah kegiatan mencatat transaksi-transaksi hingga kegiatan menutup buku, yang dapat dirinci sebagai berikut :

1. Jurnal yaitu : kegiatan mencatat transaksi-transaksi keuangan yang terjadi pada perusahaan.
2. Posting yaitu : kegiatan pembukuan catatan dari jurnal ke dalam rekening buku besar yang bersangkutan.
3. Neraca saldo (*trial balance*) yaitu : kegiatan menguji kebenaran saldo -saldo debit dan kredit rekening buku besar dengan cara menyusun saldo -saldo rekening buku besar ke dalam suatu daftar yang disebut neraca saldo.
4. Ayat penyesuaian (*adjusting entries*) yaitu : kegiatan menyesuaian jumlah-jumlah yang ada pada necara saldo, yang belum sesuai, sehingga jumlah-jumlah tersebut sesuai dengan keadaan yang sebenarnya pada akhir periode.
5. Laporan keuangan (*financial statement*) yaitu : kegiatan menyusun neraca (*balance sheet*), laporan laba - rugi (*income statement*), dan laporan sisa laba berdasarkan data - data neraca saldo yang telah disesuaikan.
6. Ayat penutup (*clossing entries*) yaitu : kegiatan menyusun pos - pos penutup. (F. Winarni ; 2011 : 27-28).

II.1.4. Siklus Akuntansi

Siklus Akuntansi adalah urutan kerja yang harus dibuat oleh akuntan, sejak awal hingga menghasilkan laporan keuangan suatu perusahaan.



Gambar II.1. Siklus Akuntansi

Sumber : Rudianto ; 2009 : 14

Keterangan gambar :

a. Dokumen Dasar

   Adalah bukti transaksi yang dijadikan dasar oleh akuntan untuk mencatat, seperti : faktur, kuitansi, nota penjualan, invoice, dll.

b. Jurnal (*Journal*)

   Adalah aktivitas meringkas dan mencatat transaksi perusahaan berdasarkan dokumen dasar. Tempat untuk mencatat dan meringkas transaksi tersebut disebut Buku jurnal.

c. *Posting*

   Adalah aktivitas memindahkan catatan di buku jurnal ke dalam buku besar sesuai jenis transaksi dan nama mperkiraan masing - masing.

d. Buku besar (*General Ledger*)

Adalah kumpulan dari semua akun/ perkiraan yang dimiliki oleh perusahaan yang saling berhibungan satu sama lainnya dan merupakan suatu kesatuan.

e. Akun/ Perkiraan (*Account*)

Adalah suatu kelas informasi di dalam suatu sistem akuntansi. Atau suatu media yang digunakan untuk mencatat informasi sumber daya perusahaan dan informasi lainnya berdasarkan jenisnya. Misalnya perkiraan kas, perkiraan piutang, akun modal, dsb. (Rudianto ; 2009 : 14)

II.1.5. Sistem Informasi Akuntansi

Kumpulan beberapa sub sistem yang saling berhubungan satu sama lain dan bekerja sama dengan harmonis dalam mengolah data keuangan sedemikian rupa hingga menghasilkan informasi keuangan bagi (pihak) manajemen untuk membantu pengambilan keputusan dibidang keuangan. (Eddy Prahasta ; 2009 : 108).

II.1.6. Penjualan

Kegiatan yang terdiri dari penjualan barang atau jasa baik secara kredit maupun secara tunai. Fungsi yang terkait dalam prosedur penjualan :

a. Fungsi Kas

Fungsi ini bertanggungjawab atas sebagai penerima kas dari pembeli.

b. Fungsi Gudang

Fungsi ini bertanggungjawab atas sebagai pencatat transaksi dan penerimaan kas dan pembuatan laporan kas.

c. Fungsi Akuntansi

Fungsi ini bertanggungjawab sebagai pencatat transaksi penjualan dan penerimaan kas dan pembuatan laporan penjualan.

d. Fungsi Pengiriman

Fungsi pengiriman berfungsi untuk menyerahkan barang yang kuantitas, mutu, dan spesifikasinya sesuai dengan yang tercantum dalam faktur penjualan yang diterima dari fungsi penjualan.

Catatan akuntansi yang digunakan dalam sistem penjualan adalah :

a. Jurnal Penjualan

Catatan akuntansi ini digunakan untuk mencatat transaksi penjualan baik secara kredit maupun tunai.

b. Jurnal Umum

Catatan akuntansi ini digunakan untuk mencatat harga pokok produk yang dijual selama periode tertentu.

c. Kartu Persediaan

Catatan akuntansi ini merupakan buku pembantu yang berisi rincian mutasi setiap jenis persediaan. (Mulyadi : 2008 ; 53)

II.1.7. Laporan Keuangan

Laporan keuangan pada dasarnya adalah hasil dari proses akuntansi yang dapat digunakan sebagai alat untuk berkomunikasi antara data keuangan atau aktivitas suatu perusahaan dengan pihak- pihak yang berkepentingan dengan data atau aktivitas perusahaan tersebut (S. Munawir. Akuntan ; 2010 : 02). Adapun tujuan dari pembuatan laopran keuangan sebagai berikut :

a. Untuk memberikan informasi keuangan yang dapat dipercaya mengenai sumber-sumber ekonomi, dan kewajiban serta modal suatu perusahaan.
b. Untuk mamberikan informasi penting lainnya mengenai perubahan dalam sumber-sumber ekonomi dan kewajiban, sperti informasi mengenai aktivitas pembelanjaan (Rudianto ; 2009 : 19)

Jenis – Jenis Laporan Keuangan :

a. Laporan Laba Rugi yaitu laporan keuangan yang memberikan informasi tentang hasil kegiatan operasi perusahaan (laba atau rugi) selama satu kurun waktu (periode) tertentu.
b. Laporan Ekuitas yaitu laporan keuangan yang memberikan informasi tentang perubahan ekuitas pemilik atau modal selama kurun waktu (periode) tertentu.
c. Neraca yaitu laporan keuangan yang memberkan informasi tentang aset, kewajiban dan ekuitas perusahaan pada saat (tanggal) tertentu.
d. Laporan Arus Kas yaitu laporan keuangan yang memberikan informasi tentang penerimaan dan pembayaran kas perusahaan selama kurun waktu (periode) tertentu. (Rudianto ; 2009 : 19)

II.2. Basis Data (*Database*)

Database atau basis data adalah sekumpulan data yang memiliki hubungan secara logika dan diatur dengan seusunan tetentu serta disimpan dalam media penyimpanan komputer. Data itu sendiri adalah representasi dari semua fakta yang ada pada dunia nyata. Database sering digunakan untuk melakukan proses terhadap data - data tersebut untuk menghasilkan informasi tetentu, misalnya dari data nama siswa yang berulang tahun pada hari ini. Tentu saja informasi tersebut

akan anda dapatkan dari software pemroses database dengan cara anda memberikan perintah dalam bahasa tertentu yaitu SQL (*Structured Query Language*).

Pada era kemajuan teknologi seperti sekarang ini, nilai informasi sangatlah penting, terlebih bagi kemajuan perusahaan. Oleh karena itu penggunaan dan penguasaan database sangat penting. Dalam database ada sebutan - sebutan untuk satuan data yaitu:

1. Karakter, ini adalah satuan data terkecil. data terdiri atau susunan karakter yang pada akhirnya memawakili data yang memiliki arti dari sebuah fakta.
2. Field, adalah kumpulan dari karakter yang mewakili fakta tertentu misalnya seperti nama siswa, tanggal lahir, dan lain-lain. Dalam dunia perancangan database, field juga disebut atribut. Bila dipandang dari sudut pemrogaman berorientasi obyek maka name dan properti tipe. Properti name atau nama adalah properti dari field yang berisi field yang mewakili data sejenis yang disimpannya. Sedangkan properti tipe adalah properti yang mengatur tipe data dari data yang akan ditampungnya, misalnya nama fieldnya adalah nama siswa maka tipe datanya adalah char, bila nama fieldnya adalah tanggal lahir maka tipe datanya adalah date, field dilihat seperti kolom.
3. Record, adalah kumpulan dari field. Pada record anda dapat menemukan banyak sekali informasi penting dengan cara mengombinasikan field - field yang ada.
4. Tabel, adalah sekumpulan dari record - record yang memiliki kesamaan entity dalam dunia nyata. Kumpulan dari tabel adalah database, wujud fisik sebuah

database dalam komputer adalah sebuah file yang didalamnya terdapat berbagai tingkatan data yang telah disebutkan di atas.

5. File, adalah bentuk fisik dari penyimpanan data. File database berisi semua data yang telah disusun dan diorganisasikan sedemikian rupa sehingga memudahkan pemberian informasi (Wahana Komputer : 2010 : 24).

II.3. Entity Relationship Diagram

Pada dasarnya ERD (*Entity Relationship Diagram*) adalah sebuah diagram yang secara konseptual memetakan hubungan antar penyimpanan pada diagram DFD di atas. ERD ini digunakan untuk melakukan pemodelan terhadap struktur data dan hubungannya. Penggunaannya ERD ini dilakukan untuk mengurangi tingkat kerumitan penyusunan sebuah database yang baik.

Entity dapat berarti sebuah obyek yang dapat dibedakan dengan obyek lainnya. Obyek tersebut dapat memiliki komponen - komponen data (atribut atau field) yang membuatnya dapat dibedakan dari obyek yang lain. Dalam dunia database *entity* memiliki atribut yang menjelaskan karakteristik dari entity tersebut. Ada dua macam atribut yang di kenal deskriptif. Hal ini berarti setiap entity memiliki himpunan yang diperlukan sebuah primary key untuk membedakan anggota - anggota dalam himpunan tersebut.

Atribut dapat memiliki sifat - sifat sebagai berikut :

1. *Atomic*, atomik adalah sifat dari atribut yang menggambarkan bahwa atribut tesebut berisi nilai yang spesifik dan tidak dapat dipecah lagi. Contoh dari sifat atomik adalah field status dari tabel karyawan yang hanya berisi menikah atau single.

2. *Multivalued,* sifat ini menandakan atribut ini bisa memiliki lebh dari satu nilai untuk tiap entity tertentu. Misalnya adalah field hobi, hobi dari tiap karyawan mungkin dan hampir pasti lebih dari satu. Misalnya karyawan A memiliki hobi membaca, menonton televisi dan bersepeda.
3. *Composite,* atribut yang bersifat komposit adalah atribut yang nilainya adalah gabungan dari beberapa atribut yang bersifat atomik. Contohnya adalah atribut alamat yang dapat dipecah menjadi atribut atomik berupa alamat, kode pos, nomor telepon, dan kota. (Wahana Komputer : 2010 : 30)

Ada beberapa derajat relasi tabg daoat terjadi, yaitu :

1. *One to one,* menggambarkan bahwa antara 1 anggota entity A hanya dapat berhubungan dengan 1 anggota entity B. Biasanya derajat relasi ini digambarkan dengna simbol 1-1.
2. *One to many*, menggambarkan bahwa 1 anggota entity A dapat memiliki hubungan dengan lebih dari 1 angota entity B. Biasanya derajat relasi ini digambarkan denga simbol 1-N.
3. *Many to many*, menggambarkan bahwa lebih dari satu anggota A dapat memiliki hubungan dengan lebih dari satu angota entity B. Simbol yang digunakan adalah N-N. (Wahana Komputer : 2010 : 31).

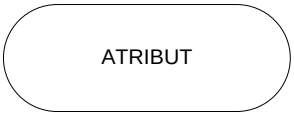
Elemen - elemen dari ERD adalah sebagai berikut :

1. *Entitas*, biasanya berupa orang, kejadian, atau benda dimana data akan dikumpulkan. Untuk menjadi entitas suatu objek harus menampilkan beberapa kali *event*.

Gambar II.2. Lambang Entity

Sumber : Kusrini ; 2009 : 21

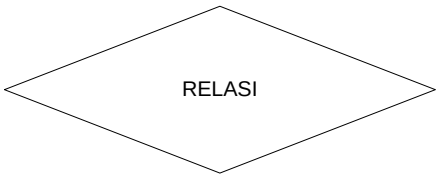
2. *Attribut*, informasi yang diambil tentang sebuah *entitas*, hanya digunakan oleh organisasi yang dimasukkan dalam model, nama atribut harus merupakan kata benda, kadang nama *entitas* diletakkan di depan nama *atribut* untuk ketelitian.



Gambar II.3. Lambang Atribut

Sumber : Kusrini ; 2009 : 22

3. *Identifier*, satu atau lebih atribut dapat menjadi identifier entitas, yang secara unik mengidentifikasi setiap anggota dari entitas, *identifier* gabungan terdiri dari beberapa atribut, bisa saja artifisial, seperti membuat nomor ID, dan tidak akan dikembangkan sampai fase desain.
4. *Relationship,* hubungan antar entitas, entitas pertama dalam *relationship* disebut entitas induk, entitas kedua disebut entitas anak. *Relationship* harus memiliki nama berupa kata kerja, *relationship* berjalan dua arah.



Gambar II.4. Lambang Relasi

Sumber : Kusrini ; 2009 : 21

5. *Kardinalitas,* mengacu pada berapa kali *instance* dari suatu entitas.
6. *Modalitas,* mengacu apakah suatu *instance* dari entitas anak dapat ada tanpa suatu telasi dengan *instance* induk atau tidak. (Hanif Al Fatta ; 2010: 121-127) .

II.3.1. Normalisasi

Setelah melalui tahapan di atas atau ERD, maka hasil pada diagram tersebut mulai direlasasikan pada tabel - tabel database. Untuk itu dperlukan sebuah tahapan yang disebut normalisasi. Normalisasi data adalah proses di mana tabel - tabel pada database dites dalam hal salingtergantungan di antara field - field pada sebuah tabel. Misalnya jika pada sebuah tabel terdapat ketergantungan terhadap lebih dari satu field dalam tabel tersebut, maka tabel tersebut harus dipecah menjadi banyak tabel.

Pada prose normalisasi data, aturan yang dijadikan acuan adalah metode ketergantungan fungsional. Teorinya adalah bahwa tiap kolom dalam sebuah tabel selalu memiliki hubungan yang unik dengan sebuah kolom kunci. Misalnya pada sebuah tabel data_siswa ada field nomor induk data field nama siswa serta field

tanggal lahir. Maka ketergantungan fungsionalnya dapat dinyatakan sebagai berikut : nmr_induk -> nm_siswa dan nmr_induk -> tgl_lahir. Artinya nm_siswa memiliki ketergantungan fungsional terhadap nmr_induk. Field nm_siswa isinya juga ditentukan oleh field nmr_induk. Maksud dari semua itu adalah nmr_induk adalah field kunci yang menentukan karena tidak ada nomor induk yang sama pada satu sekolah, jadi field nmr_induk dapat dijadikan patokan untuk mengisi nm_siswa dan field lainnya.

Ada beberapa langkah dalam normalisasi tabel, yaitu :

1. *Decomposition*, dekomposisi adalah proses mengubah bentuk tabel supaya memenuhi syarat tertentu sebagai tabel yang baik. Dekomposisi dapat dikatakan berhasil jika tabel yang dikenal dekomposisi bila digabungkan kembali dapat menjadi tabel awal sebelum didekomposisi. Dekomposisi akan sering dilakukan dalam proses normalisasi untuk memenuhi syarat - syaratnya
2. Bentuk tidak normal, pada bentuk ini semua data yang ada pada tiap entity (diambil atributnya) masih ditampung dalam satu tabel besar. Data yang ada pada tabel ini masih ada yang redundansi dan ada juga yang kosong. Semuanya masih tidak tertata rapi.
3. Normal Form Pertama (1^{st} Normal Form), pada tahapan ini tabel didekomposisi dari tabel bentuk tidak normal yang kemudian dipisahkan menjadi tabel - tabel kecil yang memiliki kriteria tidak memiliki atribut yang bernilai ganda dan komposit. Semua atribut harus bersifat atomik.

4. Normal Form Kedua (2nd Normal Form), pada tahapan ini tabel dianggap memenuhi normal kedua jika pada tabel tersebut semua atribut yang bukan kunci primer bergantung penuh terhadap kunci primer tabel tersebut .
5. Normal Form Ketiga (3rd Normal Form), setiap atribut pada tabel selain kunci primer atau kunci utama harus bergantung penuh pada kunci utama. Bentuk normal ketiga biasanya digunakan bila masih ada tabel yang belum efisien. Biasanya penggunaan bentuk normal (normalisasi) hanya sampai pada bentuk ketiga, dan tabel yang dihasilkan telah memiliki kualitas untuk membentuk sebuah database yang dapat diandalkan. Semua tabel diatas juga telah memenuhi bentuk normal tahap ketiga. (Wahana Komputer : 2010 : 32)

II.4. Pemrograman Visual Basic 2008

Visual basic 2008 merupakan salah satu bagian dari produk pemograman terbaru yang dikeluarkan oleh Microsoft, sebagai produk lingkungan pengembangan terintegritasi atau IDE andalan yang dikeluarkan. Visual basic 2010 menambahkan perbaikan – perbaikan filtur dan filtur baru yang lebih lengkap dibandingkan versi visual studio pendahulunya yaitu VB 2008.

Visual studio merupakan produk pemograman andalan dari Microsoft corporation, yang didalamnya berisi beberapa jenis IDE pemrograman seperti visual basic , visual C ++, visual Web developer, Visual C#, dan Visual f#. semua IDE pemrograman tersebut sudah mendukung penuh implementasi. Net Framework terbaru, yaitu Net Framework 4.0 yang merupakan pengembangan

dari Net Framework 3.5. Adapun database standar yang disertakan adalah Microsoft SQL Server 2008 Express. (Andi Yogyakarta : 2010 : 2).

II.5. SQL Server 2008

SQL Server 2008 adalah sebuah terobosan baru dari Microsoft dalam bidang database. SQL Server adalah sebuah *DBMS (Database Management System)* yang dibuat oleh *Microsof*t untuk ikut berkecimpung dalam persaingan dunia pengolahan data menyusul pendahulunya seperti IBM dan *Oracle. Sql server 2008* dibuat pada saat kemajuan dalam bidang hardware sedemikian pesat. Oleh karena itu sudah dapat dipastikan bahwa *SQL Server 2008* membawa beberapa terobosan dalam bidang pengolahan dan penyimpanan data. (Wahana Komputer ; 2010 : 5).

SQL server 2008 termasuk salah satu mesin database relasional. Ide tentang sistem database relasional pertama kali dicetuskan oleh seorang yang bernama E.F. Codd lewat sebuah artikel yang berjudul " *A Relation Model of Data for Large Shared Data Banks"*. Artikel tersebut ditulis pada tahun 1970. Sistem database relational berdasarkan pada metode matematika. Sistem ini menggantikan metode lama yang berdasarkan *network* dan *hierarki*. Metode relasional ini bekerja berdasarkan keterkaitan antartabel. (Wahana Komputer ; 2010 : 25).

II.6. UML

Unified Modeling Language (UML) adalah keluarga notasi grafis yang didukung oleh meta-model tunggal, yang membandu pendeskripsian dan desain

sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek (OOP).

Bahasa pemodelan grafis telah ada di industru perangkat lunak sejak lama. Pemicu umum dibalik semuanya adalah bahwa bahasa pemrograman berada pada tingkat abstraksi yang tidak terlalu tinggi untuk memfasilitasi diskusi tentang desain.

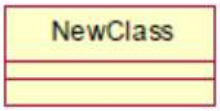
UML merupakan standar yang relatif terbuka yang dikontrol oleh *Object Management Group* (OMG), sebuah konsorium terbuka yang terdiri dari banyak perusahaan. OMG dibentuk untuk membuat standar - standar yang mendukung interoperabilitas, khususnya interoperabilitas sistem yang berorientasi objek. OMG mungkin lebih dikenal dengan standar - standar CORBA (*Common Object Request Broker Architecture*).

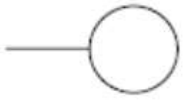
UML lahir dari penggabungan banyak bahasa permodelan grafis berorientasi objek yang berkembang pesat pada akhir 1980-an dan awal 1990-an. Sejak kehadirannya pada tahun 1997, UML menghancurkan menara Babel tersebut menjadi sejarah. (Prabowo Pudjo Widodo : 2011 : 6-9).

Dalam merancang UML (*Unified Modelling Language*) ada notasi - notasi yang telah ditentukan yaitu sebagai berikut (lihat tabel II.1) :

Tabel II.1. Notasi - notasi dalam UML

Notasi	Keterangan
Actor 	*Actor* menggambarkan segala pengguna *software* aplikasi (*user*). *Actor* memberikan suatu gambaran jelas tentang apa yang harus dikerjakan *software* aplikasi. Sebagai contoh sebuah actor dapat memberikan input kedalam dan menerima informasi dari *software* aplikasi, perlu dicatat bahwa sebuah *aktor* berinteraksi dengan *use case*, tetapi tidak memiliki kontrol atas *use case*. Sebuah *actor* mungkin seorang manusia, satu *device*, *hardware* atau sistem informasi lainnya.
Use Case 	*Use Case* menjelaskan urutan kegiatan yang dilakukan *actor* dan sistem untuk mencapai suatu tujuan tertentu. Walaupun menjelaskan kegiatan, namun *use case* hanya menjelaskan apa yang dilakukan oleh *actor* dan sistem bukan bagaimana *actor* dan sistem melakukan kegiatan tersebut. 1. *Use Case* Konkret adalah *use case* yang dibuat langsung karena keperluan *aktor*. *Aktor* dapat melihat dan berinisiatif terhadapnya . 2. *Use Case* Abstrak adalah *use case* yang tidak pernah berdiri sendiri. *Use Case* abstrak senantiasa termasuk didalam (*include*), diperluas dari (*extend*) atau memperumum (*generalize*) *use case* lainnya. Untuk menggambarkannya dalam *use case* model biasanya digunakan *association*

	relationship yang memiliki *stereotype include*, *extend* atau *generalization relationship*. Hubungan *include* menggambarkan bahwa suatu *use case* seluruhnya meliputi fungsionalitas dari *use case* lainnya. Hubungan *extend* antar *use case* berarti bahwa satu *use case* merupakan tambahan fungsionalitas dari *use case* yang lain jika kondisi atau syarat tertentu terpenuhi.
Class 	*Class* merupakan pembentuk utama dari sistem berorientasi obyek, karena *class* menunjukkan kumpulan obyek yang memiliki atribut dan operasi yang sama. *Class* digunakan untuk mengimplementasikan *interface*. *Class* digunakan untuk mengabstraksikan elemen - elemen dari sistem yang sedang dibangun. *Class* bisa merepresentasikan baik perangkat lunak maupun perangkat keras, baik konsep maupun benda nyata. Notasi *class* berbentuk persegi panjang berisi 3 bagian : persegi panjang paling atas untuk nama *class*, persegi panjang paling bawah untuk operasi, dan persegi panjang ditengah untuk atribut. Atribut digunakan untuk menyimpan informasi. Nama atribut menggunakan kata benda yang bisa dengan jelas merepresentasikan informasi yang tersimpan didalamnya. Operasi menunjukkan sesuatu yang bisa dilakukan oleh obyek dan menggunakan kata kerja.
Interface	*Interface* merupakan kumpulan operasi tanpa implementasi dari suatu *class*. Implementasi operasi dalam *interface* dijabarkan oleh operasi didalam *class*.

	Oleh karena itu keberadaan *interface* selalu disertai oleh *class* yang mengimplementasikan operasinya. *Interface* ini merupakan salah satu cara mewujudkan prinsip enkapsulasi dalam obyek.
Interaction	*Interaction* digunakan untuk menunjukkan baik aliran pesan atau informasi antar obyek maupun hubungan antar obyek. Biasanya *interaction* ini dilengkapi juga dengan teks bernama operation *signature* yang tersusun dari nama operasi, parameter yang dikirim dan tipe parameter yang dikembalikan.
Note 	*Note* digunakan untuk memberikan keterangan atau komentar tambahan dari suatu elemen sehingga bisa langsung terlampir dalam model. *Note* ini bisa disertakan ke semua elemen notasi yang lain.
Dependency	*Dependency* merupakan relasi yang menunjukan bahwa perubahan pada salah satu elemen memberi pengaruh pada elemen lain. Elemen yang ada di bagian tanda panah adalah elemen yang tergantung pada elemen yang ada dibagian tanpa tanda panah. Terdapat 2 *stereotype* dari *dependency*, yaitu *include* dan *extend*. *Include* menunjukkan bahwa suatu bagian dari elemen (yang ada digaris tanpa panah) memicu eksekusi bagian dari elemen lain (yang ada di garis dengan panah). *Extend* menunjukkan bahwa suatu bagian dari elemen di garis tanpa panah bisa disisipkan kedalam elemen yang ada di garis dengan panah.
Association	*Association* menggambarkan navigasi antar *class*

	(*navigation*), berapa banyak obyek lain yang bisa berhubungan dengan satu obyek (*multiplicity* antar *class*) dan apakah suatu *class* menjadi bagian dari *class* lainnya (*aggregation*). *Navigation* dilambangkan dengan penambahan tanda panah di akhir garis. *Bidirectional Navigation* menunjukkan bahwa dengan mengetahui salah satu *class* bisa didapatkan informasi dari *class* lainnya. Sementara *Uni Directional Navigation* hanya dengan mengetahui *class* diujung garis *association* tanpa panah kita bisa mendapatkan informasi dari *class* di ujung dengan panah, tetapi tidak sebaliknya. *Aggregation* mengacu pada hubungan has-a , yaitu bahwa suatu *class* memiliki *class* lain, misalnya Rumah memiliki *class* Kamar.
Generalization 	*Generalization* menunjukkan hubungan antara elemen yang lebih umum ke elemen yang lebih spesifik. Dengan *generalization*, *class* yang lebih spesifik (*subclass*) akan menurunkan atribut dan operasi dari *class* yang lebih umum (*superclass*) atau *subclass* is *superclass* . Dengan menggunakan notasi *generalization* ini, konsep *inheritance* dari prinsip hirarki dapat dimodelkan.
Realization 	*Realization* menunjukkan hubungan bahwa elemen yang ada di bagian tanpa panah akan merealisasikan apa yang dinyatakan oleh elemen yang ada di bagian dengan panah. Misalnya *class* merealisasikan *package*, *component* merealisasikan *class* atau *interface*.

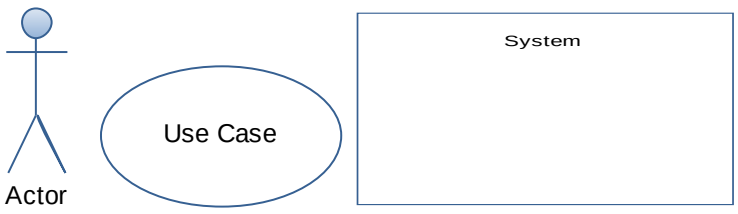
Sumber : Rosa A.S, M. Shalahuddin ; 2011 : 123 – 139

II.6.1. Diagram - Diagram UML

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

1. Use Case Diagram

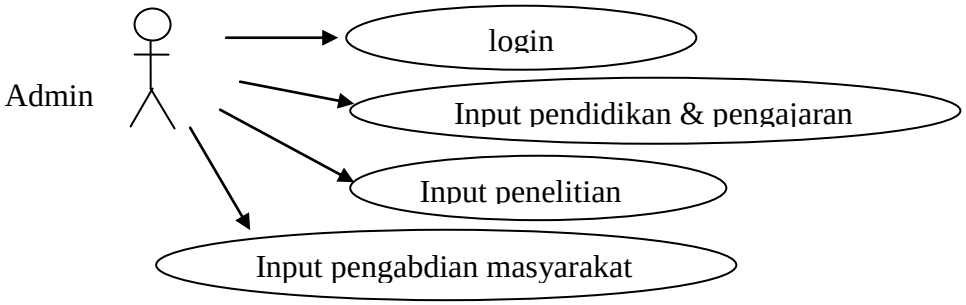
Use-case adalah konstruksi untuk mendeskripsikan bagaimana sistem akan terlihat di mata pengguna potensial. Use-case terdiri dari sekumpulan skenario yang dilakukan oleh seorang aktor (orang,perangkat keras,urutan waktu atau sistem yang lain). Sedangkan use-case diagram memfasilitasi komunikasi di antara analis dan pengguna serta diantara analis dan klien. Diagram use case menunjukkan 3 aspek dari sistem yaitu : actor, use-case, dan system boundary. Actor adalah pengguna sistem, biasa nya mewakili peran orang, sistem yang lain atau alat yang berkomunikasi dengan use-case. Use Case adalah tugas yg dilakukan oleh actor. Sekumpulan use-case biasa nya dikelompokkan dalam suatu group yang disebut System Boundary. (Radiant Victor Imbar dan Yuliusman Kurniawan : 2012 : 4. Simbol usecase ditunjukkan pada ganbar II.5.



Gambar II.5. *Actor*

Sumber : Prabowo Pudjo Widodo dan Herlawati : 2010 : 20.

Perhatikan contoh berikut :



Gambar.II.6. Contoh Use Case Diagram

Sumber : Yuni Sugiarto : 2013 : 45

2. Activity Diagram

Activity diagrams menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, decision yang mungkin terjadi, dan bagaimana mereka berakhir. Activity diagram juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi (Radiant Victor Imbar dan Yuliusman Kurniawan : 2012 : 5).

Simbol	Keterangan
	Titik Awal
	Titik Akhir
	Activity
	Pilihan untuk pengambilan keputusan

Tabel II.2. Simbol-simbol pada Activity Diagram

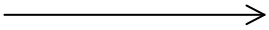
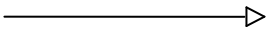
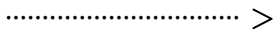
Sumber :Radiant Victor Imbar dan Yuliusman Kurniawan : 2012 : 5).

3. Class Diagram

Class diagram membantu dalam visualisas istruktur kelas-kelas dari suatu sistem dan merupakan tipe diagram yang palingbanyak. Class diagram memperlihatkan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain(dalam logical view) dari suatu sistem. Selama proses analisis, class diagram memperlihatkan aturan-aturan dan tanggung jawab entitas yang menentukan perilaku sistem. Selama proses analisis, class diagram memperlihatkan aturan-aturan dan tanggung jawab entitas yang menentukan perilaku sistem. Selama tahap decían, class diagram berperan dalam menangkap struktur dari semua kelas yang membentuk arsitektur sistem yang dibuat. Class diagram juga merupakan pondasi untuk component diagram dan deployment diagram. (Prastuti Sulistyorini : 2009 : 4).

Class diagram menggambarkan struktur statis dari kelas dalam sistem anda dan menggambarkan atribut, operasi dan hubungan antara kelas. Class diagram membantu dalam memvisualisasikan struktur kelas-kelas dari suatu sistem dan merupakan tipe diagram yang paling banyak dipakai. Selama tahap desain, class diagram berperan dalam menangkap struktur dari semua kelas yang membentuk arsitektur sistem yang dibuat. (Haviluddin : 2011 : 3).

Simbol	**Deskripsi**
Kelas **nama_kelas** +atribut +operasi()	Kelas Pada struktur sistem

Antarmuka / *interface*  nama_interface	Sama dengan konsep *interface* dalam pemrograman berorientasi objek
Asosiasi / *association* 	Relasi antar kelas dengan maksna umum, asosiasi biasanya juga disertai dengan *multiplicity*
Asosiasi berarah / *directed association* 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya disertai dangan *multiplicity*
generalisasi 	Relasi antar kelas dengan makna generalisasi – spesialisasi (umum khusus)
Keberuntungan /*dependency* 	Relasi antar kelas dengan makna kebergantungan antar kelas
Agregasi /*aggregation*	Relasi antar kelas dengan makna

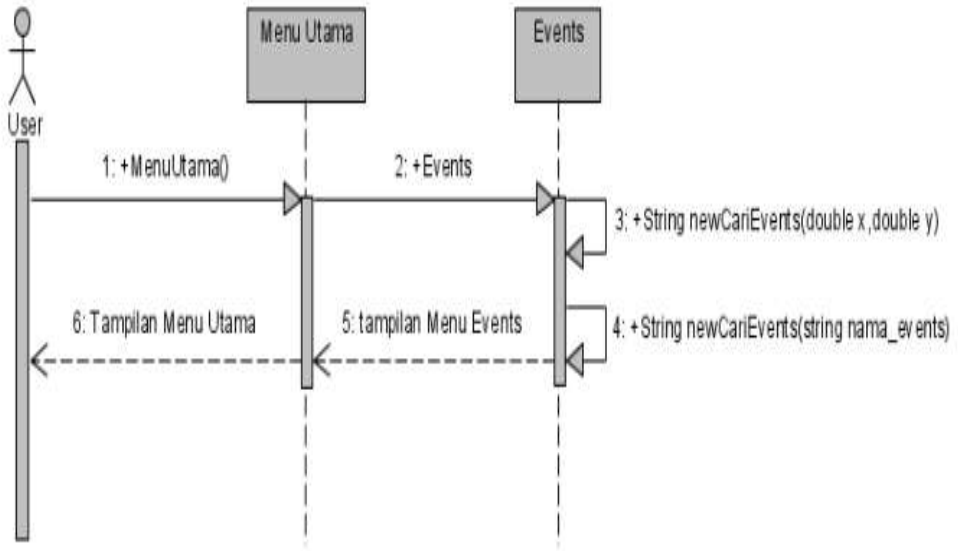
Tabel II.3. Contoh Simbol Class Diagram

Sumber : Rosa A.S dan M. Shalahuddin : 2011 : 123

4. Squence diagram

Diagram sequence menjelaskan interaksi objek yang disusun dalam suatu urutan waktu. Diagram ini secara khusus berasosiasi dengan usecase. Sequence diagram memperlihatkan tahap demi tahap apa yang seharusnya terjadi untuk

menghasilkan sesuatu didalam use case. Diagram sequence sebaiknya digunakan diawal tahap desain atau analisis karenakesederhanaannya dan mudah untuk dimengerti. Sequence diagram menjelaskan interaksi objek yang disusun berdasarkan urutan waktu. Secara mudahnya sequence diagram adalah gambaran tahap demi tahap, termasuk kronologi(urutan) perubahan secara logis yang seharusnya dilakukan untuk menghasilkan sesuatu sesuai dengan usecase diagram. (Haviluddin : 2011 : 5)



Gambar II.7. Contoh Squence Diagram

Sumber : S. Nofan Maulana Rachman : 2012 : 9

II.7. Kamus Data

Kamus data atau data dictionary adalah catalog fakta tentang data dan kebutuhan – kebutuhan informsai dari suatu sistem informasi. Dengan menggunakan kamus data analisis sistem yang mendefenisikan data yang

mengalir disistem dengan lengkap. Kamus data dibuat dengan tahap analisa dan digunakan baik pada tahap analisis maupun pada tahap perancangan sistem. (Octaviani, HS ; 2010 : 30).