BAB II

TINJAUAN PUSTAKA

II.1. Sistem

Definisi sistem secara umum adalah kumpulan dari bagian - bagian yang bekerja sama untuk mencapai tujuan yang sama atau sekumpulan objek - objek yang saling berelasi dan berinteraksi serta hubungan antar objek bisa dilihat sebagai satu kesatuan yang dirancang untuk mencapai satu tujuan.(Samiaji; 2009:11)

Dengan demikian, secara sederhana sistem dapat diartikan sebagai suatu kumpulan atau himpunan dari unsur atau variabel - variabel yang saling terorganisasi, saling berinteraksi dan saling bergantung satu sama lain. Murdick dan Ross (2013) mendefinisikan sistem sebagai seperangkat elemen yang digabungkan satu dengan lainnya untuk suatu tujuan bersama.(Hanif; 2007:3)

Suatu sistem mempunyai karakteristik sebagai berikut :

1. Komponen sistem (*Component*)

   Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi yang artinya saling bekerja sama membentuk suatu kesatuan. Komponen - komponen sistem dapat berupa suatu subsistem atau bagian dari sistem. Setiap sistem tidak peduli betapapun kecilnya selalu mengandung komponen - komponen.

2. 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.

3. Lingkungan luar sistem (*Environment*)

   Lingkungan luar suatu sistem adalah apapun diluar batas dari sistem yang mempengaruhi operasi sistem. Lingkungan luar sistem dapat bersifat menguntungkan dan dapat juga bersifat merugikan sistem tersebut. Lingkungan luar yang menguntungkan merupakan energi dari sistem dan dengan demikian harus tetap dijaga dan dipelihara.

4. Penghubung sistem (*Interface*)

   Penghubung merupakan media penghubung antara satu subsistem dengan yang lainnya. Melalui penghubung ini memungkinkan sumber - sumber data mengalir dari satu subsistem ke subsistem lainnya. Keluaran dari suatu sistem akan menjadi masukan untuk subsistem yang lainnya dengan melalui penghubung.

5. Masukan sistem (*Input*)

   Masukan adalah energi yang dimasukkan ke dalam sistem yang dapat berupa pemeliharaan *(maintenance input)* dan sinyal *(signal input)*. Signal input adalah energi yang diproses untuk didapat keluaran.

6. Keluaran sistem (*Output*)

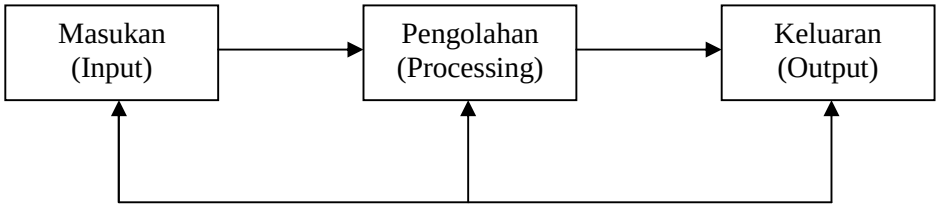
   Keluaran adalah hasil energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna. Keluaran dapat merupakan masukan untuk subsistem yang lain atau kepada supra sistem.

7. Pengolahan sistem (*Process*)

   Suatu sistem dapat mempunyai suatu proses yang akan mengubah masukan menjadi keluaran. Sistem ini akan mengolah data transaksi menjadi laporan - laporan yang dibutuhkan oleh pihak manajemen.

8. Sasaran sistem (*Objective*)

   Suatu sistem pasti mempunyai tujuan atau sasaran. Kalau suatu sistem tidak mempunyai sasaran, maka operasi sistem tidak akan ada gunanya. Suatu sistem dikatakan berhasil bila mengenai sasaran atau tujuan yang telah direncanakan. (Hanif; 2007:5-6)



Gambar II.1. Model Sistem
(*Sumber : Hanif; 2007 : 4*)

II.2. Informasi

Informasi adalah data yang sudah mengalami pemrosesan sedemikian rupa sehingga dapat digunakan penggunanya dalam membuat keputusan. Setiap

pembuatan keputusan yang rasional membutuhkan informasi sehingga memperoleh hasil yang optimal dalam kondisi pada saat keputusan tersebut dibuat. (Samiaji; 2009:12)

Jadi informasi dapat didefinisikan sebagai data yang telah diolah menjadi suatu bentuk yang berarti bagi penerima dan bermakna dalam pengambilan keputusan.

II.3. Akuntansi Biaya (*cost accounting*)

Definisi akuntansi biaya menurut Mulyadi adalah proses pencatatan, penggolongan, peringkasan, dan penyajian biaya pembuatan produk dan penjualan produk atau jasa dengan cara - cara tertentu serta penafsiran terhadapnya. Informasi yang disajikan dalam suatu laporan pembuatan dan penjualan produk tergantung kepada pemakainya. Apabila pemakai informasi tersebut adalah didalam perusahaan maka, akuntansi biaya menjadi bagian dari akuntansi manajemen. Sedangkan apabila informasi yang disajikan adalah untuk pemakai di luar perusahaan maka akuntansi biaya akan menjadi bagian dari akuntansi keuangan. Karena itu akuntansi biaya harus tunduk kepada prinsip - prinsip akuntansi yang diteruma umum yaitu Standar Akuntansi Indonesia dalam hal ini makan standar akuntansi yang digunakan adala standar untuk Indonesia. (Sampurno, 2009:1-3)

Akuntansi biaya memiliki tiga tujuan utama yaitu :

1. Menentukan harga pokok. Pada akuntansi keuangan, akuntansi biaya akan menyajikan informasi harga pokok dalam biaya historis, sedangkan pada

akuntansi manajemen, akuntansi biaya tidak terikan kepada aturan tertentu tetapi disesuaikan dengan kebutuhan manajemen di dalam perusahaan.

2. Mengendalikan biaya. Proses pengedalian biaya diawali dengan menentukan biaya yang harus dikeluarkan untuk memproduksi satu satuan produk, kemudia akuntansi biaya melakukan analisis terhadap penyimpangan dari biaya sesungguhnya dengan yang direncankan.
3. Pengambilan keputusan. Akuntasi biaya akan menyajikan informasi biaya yang berhubungan dengan masa depan, selanjutnya informasi tersebut akan menjadi acuan dalam pengambilan keputusan (Sampurno, 2009:1-4)

II.3.1. *Job Order Costing*

II.3.1.1. Pengertian *Job Order Costing*

Pengertian *Job Order Costing* menurut beberapa ahli:

1. Horngren dkk (1999) mendefinisikan *job order costing* sebagai metode mengalokasikan biaya untuk produk yang dapat segera diidentifikasi oleh unit individu atau batch, masing-masing memerlukan berbagai tingkat perhatian dan keterampilan.
2. Hansen dan Mowen (2006) mendefinisikan *job order costing* sebagai sistem perhitungan biaya yang memungkinkan biaya dikumpulkan dan dibebankan ke unit produksi untuk setiap pekerjaan.
3. Mulyadi (2007) mendefinisikan *job order costing* sebagai metode pengumpulan kos produk/jasa yang memperlakukan setiap pesanan sebagai suatu unit keluaran yang unik dan membebankan activity costs ke

setiap pesanan pada saat pesanan yang bersangkutan mengkonsumsi aktivitas.

4. Carter (2009) mendefinisikan *job order costing* sebagai suatu metode perhitungan biaya di mana biaya diakumulasikan untuk setiap pesanan (setiap batch, setiap lot, atau setiap pesanan pelanggan).

Carter (2009) menjelaskan bahwa, dalam sistem perhitungan biaya berdasarkan pesanan (*job order costing* atau *job costing*), biaya produksi diakumulasikan untuk setiap pesanan (job) yang terpisah. Suatu pesanan adalah output yang diidentifikasikan untuk memenuhi pesanan pelanggan tertentu atau untuk mengisi kembali suatu item presediaan. Hal ini berbeda dengan sistem perhitungan biaya berdasarkan proses, di mana biaya diakumulasikan untuk suatu operasi atau subdivisi dari suatu perusahaan, seperti departemen.

Perhitungan biaya berdasarkan pesanan mengakumulasikan biaya bahan baku langsung, tenaga kerja langsung, dan *overhead* yang dibebankan ke setiap pesanan. Sebagai akibatnya, perhitungan biaya berdasarkan pesanan dapat dipandang dalam tiga bagian yang saling berhubungan. Akuntansi bahan baku memelihara catatan persediaan bahan baku, membebankan bahan baku langsung ke pesanan, dan membebankan bahan baku tidak langsung ke *overhead*. Akuntansi tenaga kerja memelihara akun-akun yang berhubungan dengan beban gaji, membebankan tenaga kerja langsung ke pesanan, dan membebankan tenaga kerja tidak langsung ke *overhead*. Akuntansi *overhead* mengakumulasikan biaya *overhead*, memelihara catatan terinci atas *overhead*, dan membebankan sebagian dari *overhead* ke setiap pesanan.(Andrew & Chandra; 2010:8)

II.3.1.2. Karakteristik *Job Order Costing*

Karakteristik perusahaan yang produksinya berdasarkan pesanan dalam mengolah bahan baku menjadi produk jadi berdasarkan pesanan dari luar atau dari dalam perusahaan adalah sebagai berikut:

1. Proses pengolahan produk terjadi secara terputus-putus. Jika pesanan yang satu selesai dikerjakan, proses produksi dihentikan, dan mulai dengan pesanan berikutnya.
2. Produk dihasilkan sesuai dengan spesifikasi yang ditentukan oleh pemesan. Dengan demikian pesanan yang satu dpat berbeda dengan pesanan yang lain.
3. Produksi ditujukan untuk memenuhi pesanan, bukan untuk memenuhi persediaan di gudang.

Karakteristik usaha perusahaan yang produksinya berdasarkan pesanan berpengaruhterhadap pengumpulan biaya produksinya. Metode pengumpulan biaya produksi dengan metode harga pokok pesanan yang digunakan dalam perusahaan yang produksinya berdasarkan pesanan memiliki karakteristik sebagai berikut:

1. Perusahaan memproduksi berbagai macam produk sesuai dengan spesifikasi pemesan dan setiap jenis produk perlu dihitung harga pokok produksinya secara individual.
2. Biaya produksi harus digolongkan berdasarkan hubungannya dengan produk menjadi dua kelompok berikut ini: biaya produksi langsung dan biaya produksi tidak langsung.

3. Biaya produksi langsung terdiri biaya bahan baku dan biaya tenaga kerja langsung, sedangkan biaya produksi tidak langsung disebut dengan istilah *overhead* pabrik.
4. Biaya produksi langsung diperhitungkan sebagai harga pokok produksi pesanan tertentu berdasarkan biaya yang sesungguhnya terjadi, sedangkan biaya *overhead* pabrik diperhitungkan ke dalam harga pokok pesanan berdasarkan tarif yang ditentukan di muka.
5. Harga pokok produksi per unit dihitung pada saat pesanan selesai diproduksi dengan cara membagi jumlah biaya prduksi yang dikeluarkan untuk pesanan tersebut dengan jumlah unit prduk yang dihasilkan dalam pesanan yang bersangkutan. (Sampurno; 2009:4-5)

II.3.1.3. Manfaat *Job Order Costing*

Selanjutnya dalam perusahaan yang produksinya berdasarkan pesanan, informasi harga pokok produksi per pesanan bermanfaat bagi manajemen untuk:

1. Menentukan harga jual yang akan dibebankan kepada pemesan.
2. Mempertimbangkan penerimaan atau penolakan pesanan.
3. Memantau realisasi biaya prouksi.
4. Menghitung laba atau rugi setiap pesanan.
5. Menentukan harga pokok persedian produk jadi dan produk dalam proses yang disajikan dalam neraca.(Sampurno; 2009:4-6)

II.3.1.4. Penentuan Harga Jual

Perusahaan yang produksinya berdasarkan pesanan memproses produknya berdasark an spesifikasi yang ditentukan oleh pemesan. Dengan demikian biaya produksi pesanan yang satu akan berbeda dengan biaya produksi pesanan yang lain. Oleh karena itu harga jual yang dibebankan kepada pemesan sangat ditentukan oleh besarnya biaya produksi yang akan dikeluarkan untuk memproduksi pesanan tertentu.

Formulasi untuk menentukan harga jual yang akan dibebankan kepada pemesan adalah sebagai berikut:

Taksiran biaya produksi untuk pesanan	Rp xx
Taksiran biaya nonproduksi yang dibebankan untuk pesanan	xx +
Taksiran total biaya pesanan	Rp xx
Laba yang diinginkan	xx +
Taksiran harga jual yang dibebankan kepada pemesan	**Rp xx**

(Sampurno; 2009:4-6)

Dari formulasi tersebut akan tampak bahwa informasi taksiran biaya produksi yang akan dikeluarkan untuk memproduksi pesanan yang diinginkan oleh pemesan dipakai sebagai salah satu dasar untuk menentukan harga jual yang akan dibebankan kepada pemesan. Untuk menaksi biaya produksi yang akan dikeluarkan dalam memproduksi pesanan tertentu perlu dihitung unsur - unsur biaya berikut ini.

Taksiran biaya bahan baku	Rp xx
Taksiran biaya tenaga kerja langsung	xx
Taksiran biaya overhead pabrik	xx +
Taksiran biaya produksi	**Rp xx**

(Sampurno; 2009:4-7)

Adakalanya harga jual produk yang dipesan oleh pemesan telah terbentuk di pasar, sehingga keputusan yang perlu dilakukan oleh manajemen adalah menerima atau menolak pesanan. Untuk memungkinkan pengambilan keputusan tersebut, manajemen memerlukan informasi total harga pokok pesanan yang akan diterma tersebut. Informasi total harga pokok pesanan perusahaan memberikan dasar perlindungan bagi manajemen agar di dalam menerima pesanan perusahaan tidak mengalami kerugian. Tanpa memiliki informasi total harga pokok pesanan, manajemen tidak memiliki jaminan apakah harga yang diminta oleh pemesan dapat mendatang laba bagi perusahaan. Total harga pokok pesanan dihitung dengan unsur biaya berikut ini:

Biaya produksi pesanan:

Taksiran biaya bahan baku	Rp xx
Taksiran biaya tenaga kerja	xx
Taksiran biaya overhead pabrik	xx +
Taksiran total biaya produksi	**Rp xx**

Biaya nonproduksi:

Taksiran biaya administrasi & umum	Rp xx	
Taksiran biaya pemasaran	xx +	
Taksiran biaya nonproduksi		**Rp xx +**
Taksiran total harga pokok pesanan		**Rp xx**

(Sampurno; 2009:4-7 - 4-8)

Tahapan selanjutnya adalah informasi taksiran biaya produksi pesanan tertentu dapat dimanfaatkan sebagai salah satu dasar untuk mentepakan harga jual yang akan dibebankan kepada pemesan. Informasi taksiran biaya produksi juga bermanfaat sebagai salah satu dasar untuk mempertimbangkan diterima tidaknya suata pesanan. Jika pesanan telah diputuskan untuk diterma, manajemen memerlukan informasi biaya produksi yang sesungguhnya dikeluarkan di dalam memenuhi pesanan tertentu. Oleh karena itu, akuntansi biaya digunakan untuk mengumpulkan infomasi biaya produksi pada pesanan yang diterima untuk memantau apakah proses produksi untuk memenuhi pesanan tertentu menghasilkan total biaya produksi sesuai yang diperhitungkan sebelumnya.

Pengumpulan biaya produski per pesanan tersebut dilakukan dengan menggunakan metode harga pokok pesanan. Perhitungan biaya produksi sesungguhnya yang dikeluarkan untuk pesanan tertentu dilakukan dengan formulasi berikut ini:

Biaya bahan baku sesungguhnya	Rp xx
Biaya tenaga kerja sesungguhnya	xx
Taksiran biaya overhead pabrik	xx +
Total biaya produksi sesungguhnya	**Rp xx**

(Sampurno; 2009:4-8)

Untuk mengetahui apaka pesanan tertentu mampu menghasilkan laba bruto atau mengakibatkan rugi bruto, manajemen memerlukan informasi biaya produksi yang telah dikeluarkan untuk memproduksi pesanan tertentu. Informasi laba atau rugi bruto pada pesanan diperlukan untuk mengetahui kontribusi tiap pesanan dalam menutup biaya nonproduksi dan menghasilkan laba atau rugi. Oleh karenanya metode harga pokok pesanan digunakan oleh manajemen untuk mengumpulkan informasi biaya produksi yang sesungguhnya dikeluarkan untuk tiap pesanan guna menghasilkan informasi laba atau rugi tiap pesanan. Laba atau rugi bruto pada pesanan dihitung sebagai berikut:

Harga jual yang dibebankan kepada pemesan		**Rp xx**
Biaya produksi pesanan tertentu:		
Biaya bahan baku sesungguhnya	Rp xx	
Biaya tenaga kerja langsung sesungguhnya	xx	
Taksiran biaya overhead pabrik	xx +	
Total biaya produksi pesanan		**Rp xx -**
Laba bruto		**Rp xx**

(Sampurno; 2009:4-8 - 4-9)

II.4. *Visual Basic .NET*

Visual Basic .NET merupakan bahasa yang digunakan untuk membangun sebuah aplikasi .NET seperti halnya bahasa pemrograman lainnya. *Visual Basic* juga sangat mudah untuk dipelajari dan merupakan bahasa pemrograman yang sangat produktif.

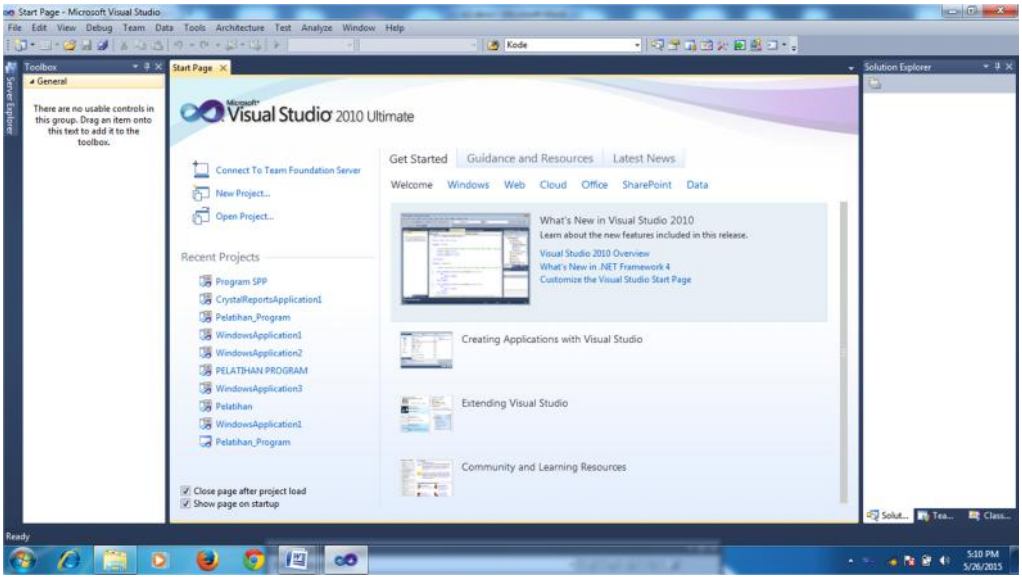
Salah satu komponen yang paling penting dalam *Visual Basic .NET* adalah *.NET Framework*. Komponen ini berisi semua tentang fungsi-fungsi sistem operasi dan dapat digunakan pada aplikasi yang akan dibuat melalui beragam *Methods*.

Bahasa *Visual Basic* 2010 awalnya berasal dari bahasa pemrograman yaitu bahasa BASIC, yang oleh *Microsoft* diadaptasi dalam program *Microsoft Quick BASIC*. Saat ini versi *Microsoft Visual Studio* yang beredar adalah versi 10 yang populer dengan nama *Microsoft Visual Studio* 2010.

Visual Studio adalah *Integrated Development Environment (IDE)* dari untuk membangun aplikasi *console* dan *Graphical user interface (GUI)* dengan menggunakan bahasa yang didukung pada *.NET Framework*. Aplikasi GUI yang dapat dibangun diantaranya adalah *Windows Form*, *Website*, *Web Application*, *Windows Mobile*.

Visual Studio selain mempunyai feature untuk :

1. Designer antarmuka untuk Winform, WPF dan Web. Selain itu juga dapat digunakan untuk mendesign Class, Data dan Mapping.
2. Code editor dengan dukungan *IntelliSense*.
3. Debugger. (M. Reza Faisal; 2013:1-5 - 1-6)



Gambar II.2. *Microsoft Visual Studio* 2010
(*Sumber : M. Reza Faisal; 2013:1-6*)

II.5. *SQL Server 2008*

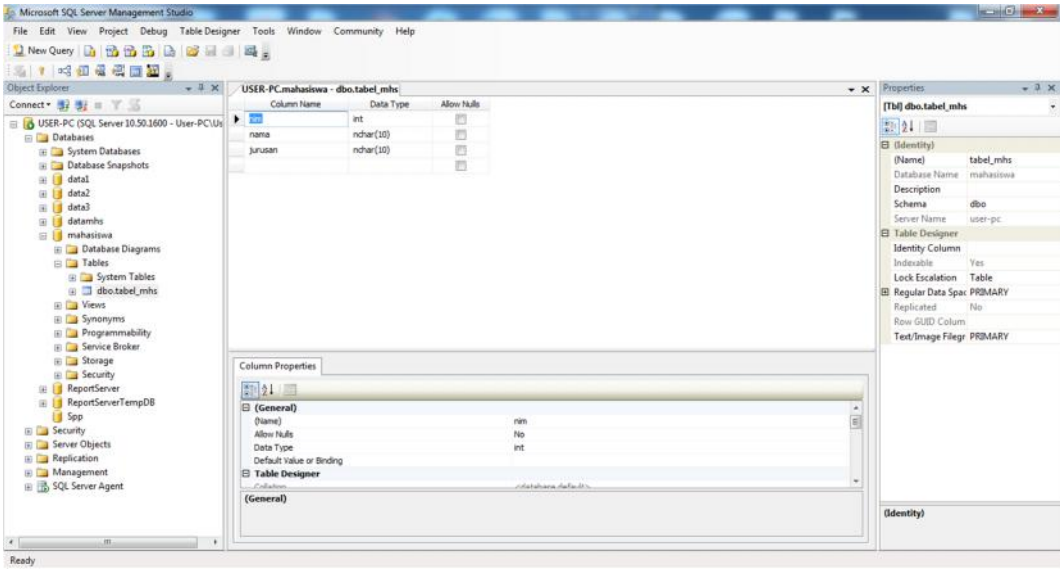
SQL Server adalah sebuah tipe database yang dinamakan database relasional (RDBMS), yaitu database yang mengorganisasikan data dalam bentuk tabel. Tabel dibentuk dengan mengelompokkan data yang mempunyai subjek yang sama yang berisi baris dan kolom informasi.

SQL Server 2008 adalah sebuah RDBMS (*Relational Database Management System*) yang di-develop oleh *Microsoft*. *SQL Server* 2008 menggunakan SQL *Language (Structur Query Language*). Sebuah database berisi satu tabel atau lebih dan memiliki nama yang berbeda untuk masing - masing tabel, memiliki *field* - *field* dan berisi *record*. *Query* digunakan untuk menyimpan dan mengolah data. Pada *SQL Server* 2008, kita bisa melakukan pengambilan dan modifikasi data yang ada dengan cepat dan efisien. Pada *SQL Server* 2008, kita bisa membuat objek - objek yang sering digunakan pada aplikasi bisnis seperti membuat

database, tabel*, function, store procedure, trigger*, dan *view*. (Cybertron Solution dan SmitDev Community;2010:101)

Gambar II.3. Login SQL Server
(*Sumber : Cybertron Solution & SmitDev Community; 2010: 102*)

Terdapat beberapa *view* yang akan sering digunakan pada *SQL Server Management Studio*, di antaranya *Object Explorer* yang digunakan untuk melakukan aktivitas pada database menggunakan GUI dan *Query* yang digunakan untuk melakukan aktivitas database menggunakan *T-SQL Query*. Pada *SQL Server* 2008, database adalah objek yang paling vital karena pada objek tersebutlah objek dan data didefinisikan dan disimpan. (Cybertron Solution dan SmitDev Community;2010,103)

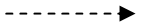
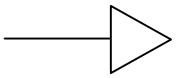
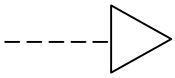
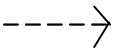


Gambar II.4. *SQL Server Management Studio*
(*Sumber : Cybertron Solution & SmitDev Community; 2010: 102*)

II.6. UML (*Unified Modelling Language)*

UML (*Unified Modelling Language*) adalah bahasa yang digunakan untuk memvisualisasikan, mendefinisikan, membangun dan membuat dokumen dari arsitektur perangkat lunak. UML dapat digunakan pada semua proses melalui metodologi pengembangan perangkat lunak dan melakukan implementasinya pada teknologi yang berbeda. (Jurnal TELEMATIKA MKOM, Vol.3 No.2, September 2011). Hal ini disebabkan karena UML menyediakan bahasa pemodelan visual yang memungkinkan bagi pengembangan sistem untuk membuat cetak biru atas visi mereka dalam bentuk yang baku, mudah dimengerti serta dilengkapi dengan mekanisme yang efektif untuk berbagi *(sharing)* dan mengkomunikasikan rancangan mereka dengan yang lain.

Tabel II.1. Relasi - Relasi dalam UML (*Unified Modelling Language*)

Relasi	Fungsi	Notasi
Asosiasi (*association*)	Mendeskripsikan hubungan antar-*instance* suatu kelas	
Ketergantungan (*dependency*)	Relasi antar dua elemen model	
Aliran (*flow*)	Relasi antar dua versi suatu objek	
Generalisasi (*generalization*)	Relasi antar penklasifikasi yang memiliki deskripsi yang bersifat lebih umum dengan berbagai perngklasifikasi yang lebih spesifik, digunakan dalam struktur pewarisan	
Realisasi (*realization*)	Relasi antar spesifikasi dan impelementasinya	
Penggunaan (usage)	Situasi di mana salah satu elemen membutuhkan elemen yang lainnya agar dapat berfungsi dengan baik	

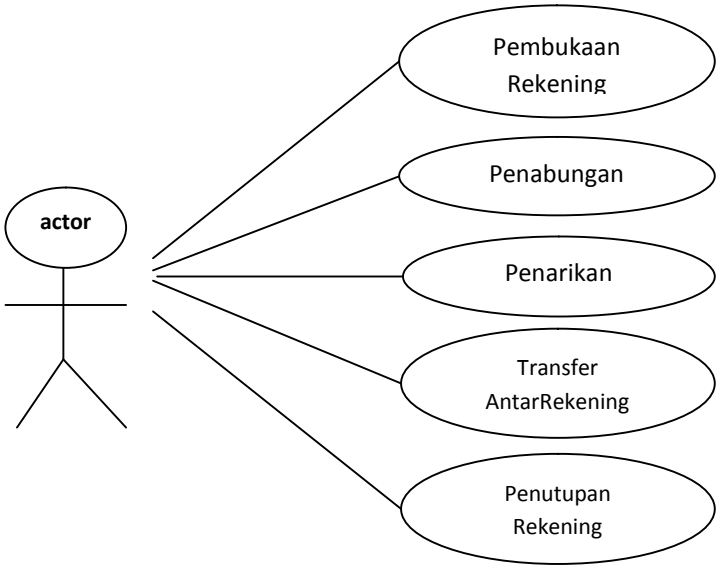
(Sumber : Adi Nugroho;2010:23)

II.6.1. *Use Case Diagram*

Use Case adalah deskripsi fungsi dari sebuah sistem dari perspektif pengguna. Use case bekerja dengan cara mendeskripsikan tipikal interaksi antara user (pengguna) sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai. Urutan langkah-langkah yang menerangkan antara pengguna dan sistem disebut scenario. Setiap scenario mendeskripsikan urutan kejadian. Setiap urutan diinisialisasikan oleh orang, sistem yang lain, perangkat keras atau urutan waktu. Dengan demikian secar singkat bisa dikatakan use case adalah serangkaian scenario yang digabungkan bersama-sama oleh tujuan umum pengguna.

Dalam konteks UML, tahap konseptualisasi dilakukan dengan pembuatan *use case diagram* yang sesungguhnya merupakan deskripsi peringkat tinggi bagaimana perangkat lunak (aplikasi) akan digunakan oleh penggunanya. Selanjutnya, *use case diagram* tidak hanya sangat penting pada saat analisis, tetapi juga sangat penting dalam tahap perancangan (*design*), untuk mencari kelas-kelas yang terlibat alam aplikasi, dan untuk melakukan pengujian (*testing*).

Saat akan mengembangkan *use case diagram*, hal yang pertama kali harus dilakukan adalah mengenali *actor* untuk sistem yang sedang dikembangkan. Dalam hal ini, ada beberapa karakteristik untuk para *actor*, yaitu *actor* yang ada di luar sistem yang sedang dikembangkan dan *actor* yang berinteraksi dengan sistem yang sedang dikembangkan. (Adi Nugroho ; 2009 : 7)

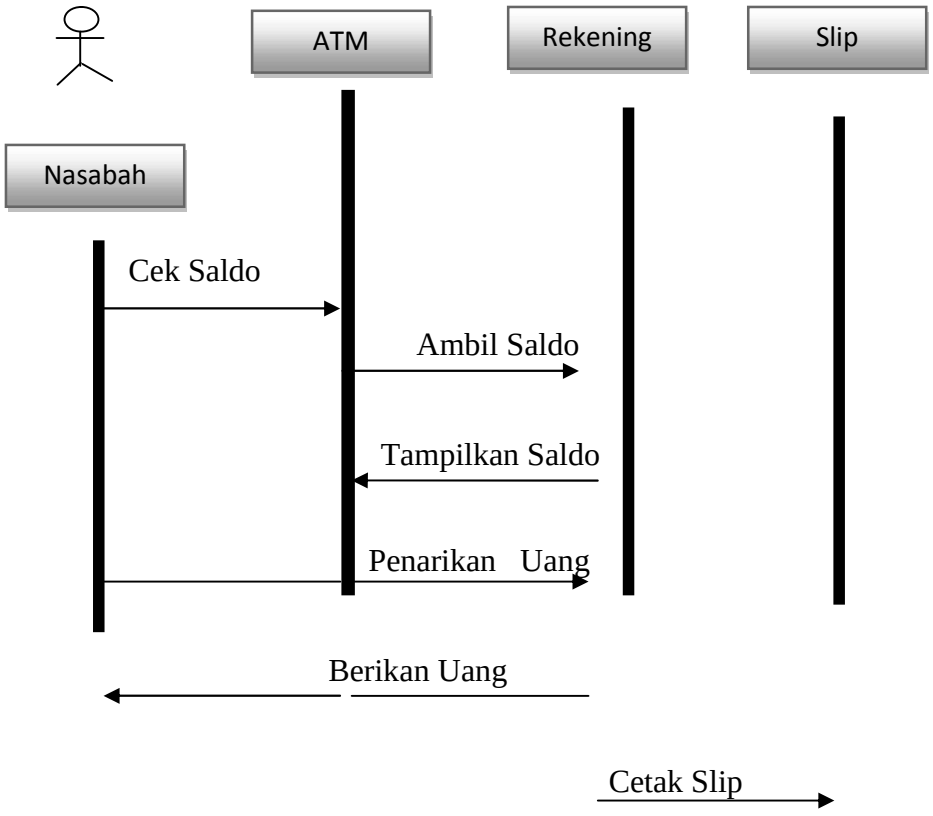


Gambar II.5. Contoh Use Case Diagram
(Sumber : Adi Nugroho ; 2009 : 8)

II.6.2. *Sequence Diagram*

Sequence diagram digunakan untuk menggambarkan perilaku pada sebuah skenario. Diagram ini menunjukkan sejumlah contoh obyek dan message (pesan) yang diletakkan diantara obyek-obyek ini di dalam use case.

Sequence diagram juga menampilkan interaksi antar suatu kelas dengan kelas yang lainnya, bagaimana suatu *message* (pesan) dikirimkan dari suatu kelas ke kelas yang lainnya, dengan penekanan lebih pada urutan kejadian menurut waktu. Keunggulan dari *Sequence diagram* memperlihatkan dengan baik urutan interaksi yang terjadi antara suatu kelas dengan kelas lainnya, tetapi mengabaikan pengorganisasiannya. (Adi Nugroho ; 2009 : 101)



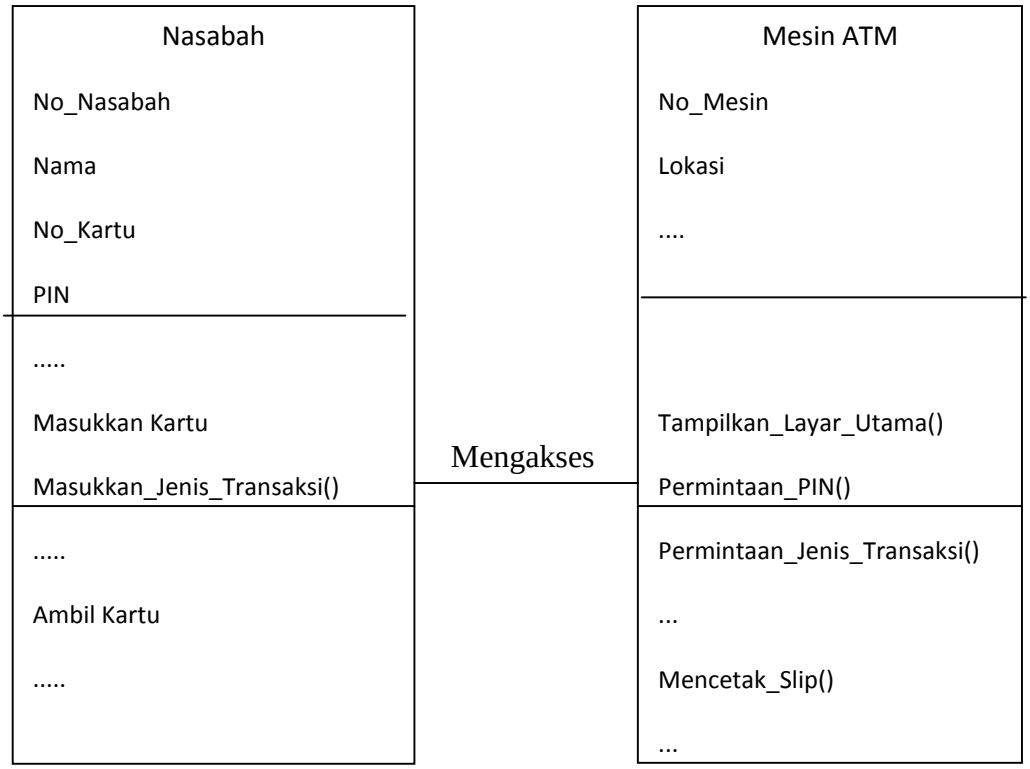
Gambar II.6. *Sequence Diagram*
(Sumber : Adi Nugroho ; 2009 : 102)

II.6.3. *Class Diagram*

Class didefenisikan sebagai kumpulan/himpunan objek yang memiliki kesamaan dalam atribut/properti, perilaku (operasi), serta cara berhubungan dengan objek lain. (Adi Nugroho ; 2009 : 18)

Selain itu, kita juga mendefenisikan objek sebagai konsep, abstraksi dari sesuatu dengan batas nyata, sehingga kita dapat menggambarkan secara sistematis. Pemahaman objek memiliki dua fungsi, yaitu :

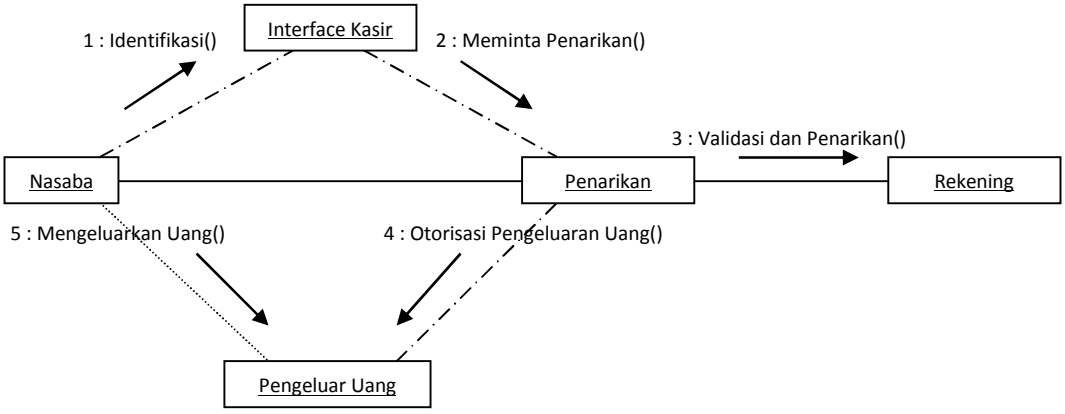
1. Memudahkan untuk mempelajari secara seksama hal-hal yang ada di dunia nyata.
2. Menyediakan suatu dasar yang kuat dalam implementasi ke dalam sistem terkomputerisasi. (Adi Nugroho ; 2009:17)



Gambar II.7. Contoh Class Diagram
(Sumber : Adi Nugroho ; 2009 : 39)

II.6.4. *Collaboration Diagram*

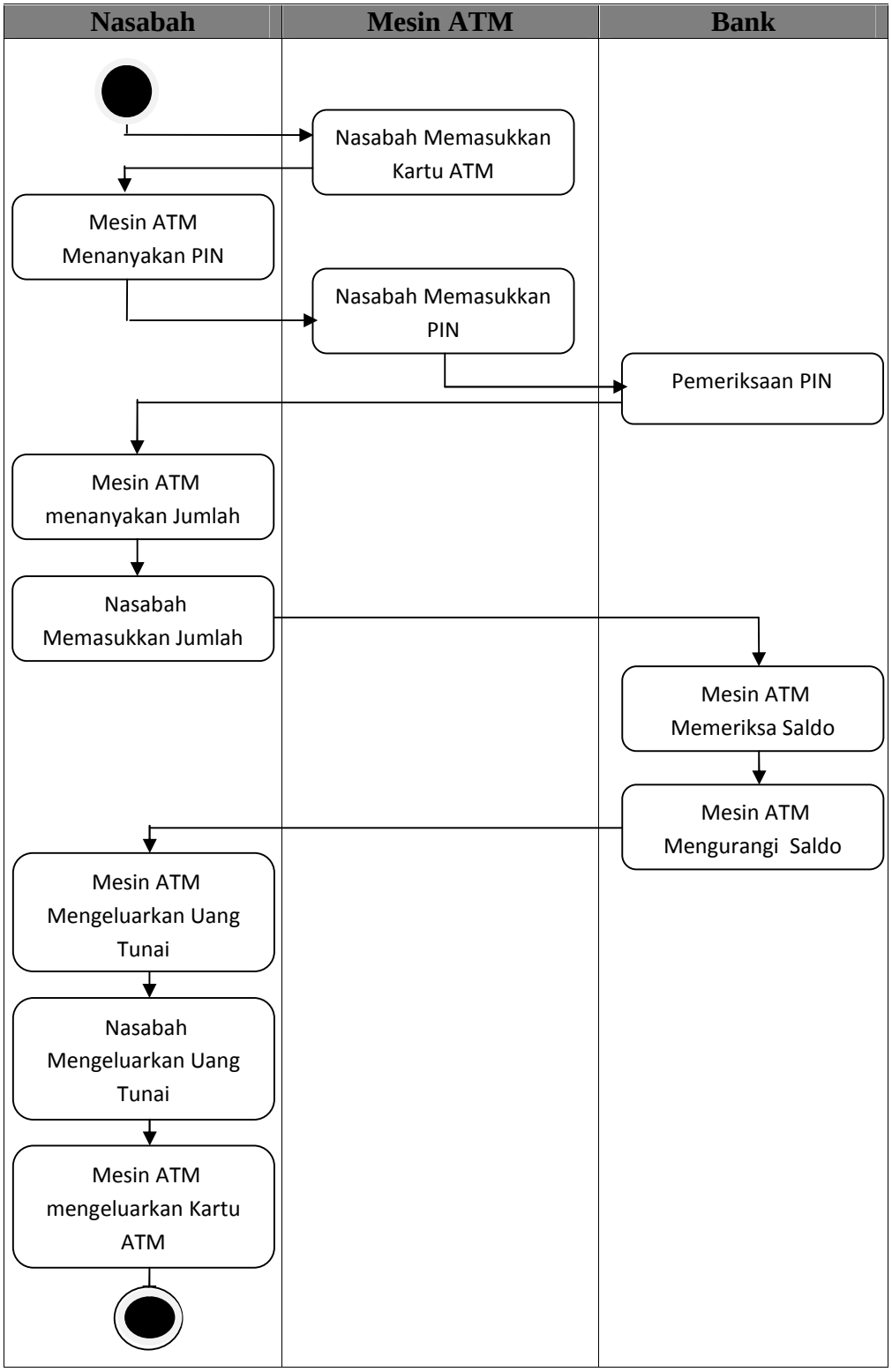
Collaboration diagram pada dasarnya merupakan diagram kelas yang memuat peran-peran pengklasifikasi dan peran-peran asosiasi, alih-alih hanya menampilkan pengklasifikasi-pengklafikasi serta asosiasi-asosiasi. Peran pengklasifikasi dan peran asosiasi mendeskripsikan konfigurasi objek-objek dan tautan-tautan yang mungkin terjadi saat suatu *instance* kolaborasi dieksekusi. (Adi Nugroho ; 2010 : 44)



Gambar II.8. Contoh Collaboration Diagram
(Sumber : Adi Nugroho ; 2010 : 44)

II.6.5. Activity View

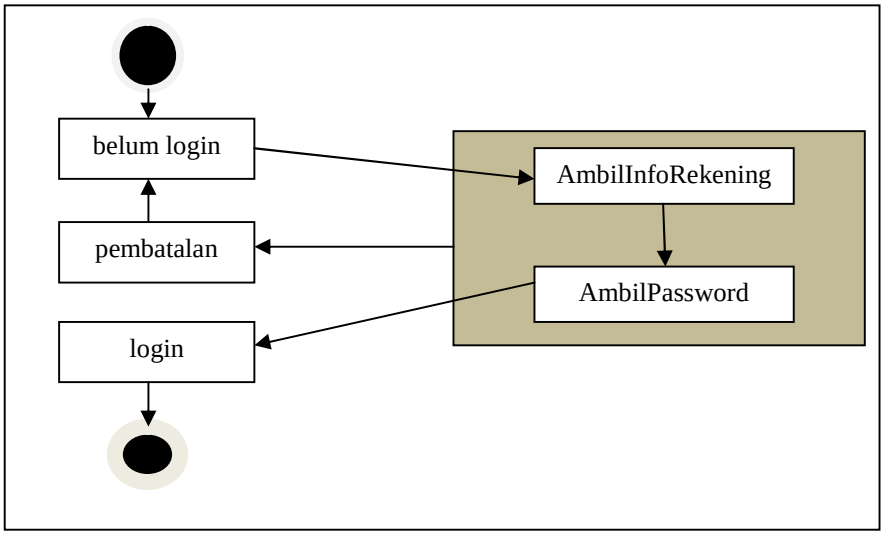
Diagram aktivitas (*activity diagram*) sesungguhnya merupakan bentuk khusus dari *state machine* yang bertujuan memodelkan komputasi-komputasi dan aliran-aliran kerja yang terjadi dalam sistem/perangkat lunak yang sedang dikembangkan. Biasanya, suatu diagram aktivitas mengasumsikan komputasi-komputasi dilaksanakan tanpa adanya interupsi-interupsi eksternal berbasis event terjadi padanya (Adi Nugroho; 2010 : 62)



Gambar II.9. Contoh Activity Diagram
(Sumber : Adi Nugroho ; 2009 : 11)

II.6.6. *Statechart Diagram*

Menggambarkan transisi dan perubahan keadaan (dari suatu state ke state lainnya) suatu objek pada sistem sebagai akibat dari stimuli yang diterima. Pada umumnya statechart diagram menggambarkan class tertentu (satu class dapat memiliki lebih dari satu statechart diagram).



Gambar II.10. *Statechart Diagram*
(Sumber : Adi Nugroho ; 2009 : 109)

II.6.7. *Component Diagram*

Component diagram menggambarkan struktur dan hubungan antar komponen piranti lunak, termasuk ketergantungan (*dependency*) di antaranya. Komponen piranti lunak modul berisi *code*, baik berisi *source code* maupun *binary code*, baik *library* maupun *executable*, baik yang muncul pada *compile time, link time*, maupun *run time*. Umumnya komponen terbentuk dari beberapa class dan/atau package, tapi dapat juga dari komponen - komponen yang lebih

kecil. Komponen dapat juga berupa interface, yaitu kumpulan layanan yang disediakan sebuah komponen untuk komponen lain.

II.6.8. *Deployment/Physical Diagram*

Menggambarkan detail bagaimana komponen di deploy dalam infrastruktur sistem, di mana komponen akan terletak (pada mesin, server atau piranti keras apa), bagaimana kemampuan jaringan pada lokasi tersebut, spesifikasi server, dan hal - hal lain yang bersifat fisikal sebuah node adalah server, workstation, atau piranti keras lain yang digunakan untuk men-deploy komponen dalam lingkungan sebenarnya. Hubungan antar node (misalnya TCP/IP) dan requirement dapat juga didefinisikan dalam diagram ini.

II.7. Basis Data (*database*)

Menurut James Martin (2009), Basis data adalah suatu kumpulan data terhubung yang disimpan secara bersama-sama pada suatu media tanpa mengatap satu sama lain atau tidak perlu suatu kerangkapan data dengan cara-cara tertentu sehingga mudah untuk digunakan dan ditampilkan kembali, dapat digunakan untuk satu atau lebih program aplikasi secara optimal, data dapat disimpan tanpa mengalami ketergantungan pada program yang akan menggunakannya, serta disimpan sedemikian rupa sehingga penambahan, pengambilan dan modifikasi data dapat dilakukan dengan mudah dan terkontrol. (Kusrini ; 2007:140)

Adapun elemen - elemen sistem manajemen basis data adalah sebagai berikut:

1. *Database*

   Database adalah kumpulan dari item data yang saling berhubungan satu sama lain, yang diorganisasikan berdasarkan sebuah skema atau struktur tertentu, tersimpan di *hardware* komputer, dan harus menggunakan *software* untuk melakukan manipulasi tertentu

2. *File*

   File adalah kumpulan *record* sejenis yang mempunyai panjang elemen dan atribut yang sama, namun *value*-nya berbeda. *Database* dibentuk dari kumpulan *file.*

3. *Record*

   Record adalah kumpulan elemen yang saling berkaitan yang menginformasikan tentang satu entitas secara lengkap. Satu *record* mewakili satu data atau informasi.

4. *Field*

   Field adalah bagian tertentu dari data dalam *record* yang mewakili satu entitas. Misalnya *file* anggota dapat dilihat dari *field*-nya. Seperti kode anggota, nama dan lain - lain.

5. *Data Value*

   Data Value adalah data actual atau informasi yang disampaikan pada setiap data elemen atau *field* data.

6. *Entity*

   Entity adalah objek riil yang dapat dibedakan satu sama lain dan tidak saling bergantung. Misal, pada bidang sirkulasi, entitasnya adalah anggota

dan buku.

7. *Query*

   Query merupakan perintah yang dirancang untuk memanggil kelompok *record* tertentu dari satu file atau lebih untuk melakukan operasi pada *file*.

8. *View*

   View adalah data yang terdiri atas sejumlah record yang diproses urutan penampilan. (Kusrini;2007:142-143)

II.8. Normalisasi

Normalisasi merupakan cara pendekatan dalam membangun desain logika basis data relasional yang tidak secara langsung berkaitan dengan model data, tetapi dengan menerapkan sejumlah aturan dan kriteria standar untuk menghasilkan struktur table yang normal. (Kusrini ; 2008 : 40)

Dalam perspektif normalisasi sebuah database dikatakan baik jika setiap table yang membentuk basis data sudah berada dalam keadaan normal. suatu tabel dikatakan normal, jika :

1. Jika ada dekomposisi/penguraian tabel, maka dekomposisinya dijamin aman (lossless-join decomposition)
2. Terpeliharanya ketergantungan functional pada saat perubahan data (dependency preservation)
3. Tidak melanggar Boyce Code Normal Form (BCNF), jika tidak bisa minimal tidak melanggar bentuk normalisasi ketiga. (Kusrini ; 2008 : 40)

II.8.1. Bentuk – Bentuk Normalisasi

1. **Bentuk Tidak Normal**

   Bentuk ini merupakan kumpulan data yang akan direkam, tidak ada keharusan mengikuti format tertentu, dapat saja tidak lengkap dan terduplikasi. data dikumpulkan apa adanya sesuai keadaannya. (Kusrini ; 2008 : 41)

2. **Bentuk Normal Pertama (1NF/First Normal Form)**

   Suatu tabel disebut 1NF jika :

   - Tidak ada baris yang duplikat dalam tabel tersebut
   - Masing – masing cell bernilai tunggal. (Kusrini ; 2008 : 41)

3. **Bentuk Normal Kedua (2NF/Second Normal Form)**

   Bentuk normal kedua terpenuhi jika pada sebuah tabel semua atribut yang tidak termasuk dalam primary key memiliki ketergantungan fungsional pada primary key secara utuh. sebuah tabel dikatakan tidak memenuhi 2NF, jika ketergantungannya hanya bersifat parsial (hanya tergantung pada sebagian dari primary key). (Kusrini ; 2008 : 41)

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

   Bentuk normal ketiga terpenuhi jika untuk setiap ketergantungan fungsional dengan notasi X->A, dimana A mewakili semua atribut tunggal di dalam tabel yang tidak ada dalam X, maka :

   - X haruslah superkey pada tabel tersebut.
   - Atau A merupakan bagian dari primary key pada tabel tersebut. (Kusrini ; 2008 : 42)

5. **Bentuk Normal Keempat Dan Kelima**

   Penerapan aturan normalisasi sampai bentuk normal ketiga sudah memadai untuk menghasilkan tabel berkualitas baik. Namun demikian, terdapat pula bentuk normal keempat (4NF) dan kelima (5NF). Bentuk normal keempat berkaitan dengan sifat ketergantungan banyak nilai (multiple dependency) pada suatu tabel yang merupakan pengembangan dari ketergantungan fungsional. Adapun bentuk normal kelima merupakan nama lain dari Project Join Normal Form (PJNF). (Kusrini ; 2008 : 43)

6. **Bentuk Normal Boyce - Codd (BCNF/ Boyce - Codd Normal Form)**

   Bentuk normal ini terpenuhi jika :

   - Memenuhi 1NF
   - Relasi harus bergantung fungsi pada atribut superkey. (Kusrini ; 2008 : 43)

II.9. Kamus Data

Kamus data (*data dictionary*) mencakup definisi - definisi dari data yang disimpan dalam basis data dan dikendalikan oleh sistem manajemen basis data. Struktur basis data yang dimuat dalam kamus data adalah kumpulan dari seluru definisi *field,* definisi tabel , relasi tabel, dan hal - hal lainnya. Nama *field* data, jenis data, nilai - nilai yang valid untuk data, dan karateristik - karakteristik

lainnya akan disimpan dalam kamus data. Perubahaan - perubahaan pada struktur data hanya dilakukan satu kali di dalam kamus data, program - program aplikasi yang mempergunakan data tidak akan ikut terpengaruh.(McLeod;2008:171).