BAB II

TINJAUAN PUSTAKA

II.1. Definisi Sistem

Menurut Kusrini (2007: 11), Sistem merupakan kumpulan elemen yang saling berkaitan yang bertanggung jawab memproses masukan(*input*) sehingga menghasilkan keluaran(*output*).

Sistem adalah sebuah tatanan yang terdiri atas sejumlah komponen fungsional (dengan tugas/fungsi khusus) yang saling berhubungan dan secara bersama-sama bertujuan untuk memenuhi suatu proses/pekerjaan tertentu (Kusrini; 2007: 11).

Ada beberapa definisi sistem, tetapi defini dari kamus *Webster's Unabridged* lebih mendekati dengan keperluan. Definisi tersebut adalah sebagai berikut: Sistem adalah elemen-elemen yang saling berhubungan membentuk satu kesatuan atau organisasi. Contohnya adalah sistem tatasurya, sistem irigasi, dan sistem informasi. Sistem tubuh manusia dengan subsistem-subsistem seperti peredaran darah, syaraf, otak, pencernaan dan sebagainya (Zulkifli Amsyah; 2005: 27).

Definisi sistem secara umum (Hanif Al Fatta; 2007: 3):

1. Kumpulan dari bagian-bagian yang bekerja sama untuk mencapai tujuan yang sama.

2. Sekumpulan objek-objek yang saling berelasi dan berinteraksi serta hubungan antara objek bisa dilihat sebagai suatu kesatuan yang dirancang untuk mencapai satu tujuan.

II.2. Keputusan

Keputusan merupakan kegiatan memilih suatu strategi atau tindakan dalam pemecahan masalah tersebut.Tindakan memilih strategi atau aksi yang diyakini manajer akan memberikan solusi terbaik atas sesuatu (Kusrini; 2007: 7).

Tujuan dari keputusan adalah untuk mencapai target atau aksi tertentu yang harus dilakukan (Kusrini; 2007: 7).

Kriteria atau ciri-ciri dari keputusan adalah:

1. Banyak pilihan/Alternatif.
2. Ada kendala atau syarat.
3. Mengikuti Suatu Pola/model tingkah laku, baik yang terstruktur maupun yang tidak terstruktur .
4. Banyak Input/variabel.
5. Ada faktor resiko.
6. dibutuhkan kecepatan, ketepatan dan keakuratan.

II.2.1. Kondisi Pengambilan Keputusan

Menurut Kusrini (2007: 9) Ada beberapa keadaan yang mungkin dialami oleh pengambil keputusan ketika mengambil keputusan yaitu:

1. Pengambilan keputusan dalam kepastian, semua alternatif diketahui secara pasti.
2. Pengambilan keputusan dalam berbagai tingkat resiko yang dipilih.

3. Pengambilan keputusan dalam kondisi ketidakpastian, ada alternatif yang tidak diketahui dengan jelas.

II.3. Sistem Pendukung Keputusan

Sistem pendukung Keputusan adalah sebuah sistem yang dibangun untuk mendukung solusi atas suatu masalah atau mengevaluasi suatu peluang. DSS yang seperti itu disebut aplikasi DSS (Decision Support Sistem). Aplikasi DSS digunakan dalam pengambilan keputusan. Aplikasi DSS menggunakan data, memberikan antarmuka pengguna yang mudah dan dapat menggabungkan pemikiran pengambil keputusan (Kusrini; 2007: 16).

Sistem pendukung keputusan (SPK) adalah suatu sistem informasi yang mengevaluasi beberapa pilihan yang berbeda guna membantu seseorang memberikan keputusan terhadap masalahnya (Sri yulianto J.P; 2008: 160).

DSS(Decision Support Sistem) tidak dimaksudkan untuk mengotomatisasikan pengambilan keputusan, tetapi memberikan perangkat interaktif yang memungkinkan pengambilan keputusan untuk melakukan berbagai analisis menggunakan model-model yang tersedia (Kusrini; 2007: 16).

Keputusan yang diambil untuk menyelesaikan suatu masalah dilihat dari keterstrukturanya yang bisa dibagi menjadi (Kusrini; 2007: 19):

1. Keputusan terstruktur (*structured decision*), yaitu keputusan yang dilakukan secara berulang-ulang dan bersifat rutin.
2. Keputusan semiterstruktur (*semistructured decision*), yaitu keputusan yang memiliki dua sifat. Sebagian keputusan bisa ditangani oleh komputer, dan yang lain tetap harus dilakukan oleh pengambil keputusan.

3. Keputusan tak terstruktur (*unstructured decision*), yaitu keputusan yang penangananya rumit karena tidak terjadi berulang-ulang atau tidak selalu terjadi.

II.4. Pinjaman/Kredit

Perkataan kredit berasal dari kata "*credo*" yang bearasal dari bahasa latin yang berarti "saya percaya", yang merupakan kombinasi dari bahasa Sanskerta yaitu *cred* yang artinya "kepercayaan", dan bahasa latin *do* yang artinya "saya tempatkan". Memperoleh kredit/pinjaman berarti memperoleh kepercayaan. Atas dasar kepercayaan kepada seseorang yang memerlukanya maka diberikan uang, barang atau jasa dengan syarat membayar kembali atau memberikan penggantianya dalam jangka waktu yang telah diperjanjikan. Dalam kehidupan sehari-hari, kredit diartikan sebagai "Pinjaman" atau "Utang" (Iswi hariyani; 2010: 9).

Pengertian kredit/pinjaman menurut menurut UU 10/1998 tentang perbankan, pasal 1 angka 11, adalah "penyediaan uang atau tagihan yang dapat dipersamakan dengan itu, berdsarkan persetujuan atau kesepakatan pinjam-meminjam antara bank dan pihak lain yang mewajibkan pihak peminjam untuk melunasi utangnya setelah jangka waktu tertentu dengan pemberian bunga" (Iswi hariyani; 2010: 10).

Unsur kredit/pinjaman yang paling esensial adalah "kepercayaan" dari kreditor terhadap peminjam. Kepercayaan tersebut timbul karena dipenuhi segala ketentuan dan persyaratan untuk memperoleh pinjaman oleh peminjam, antara

lain: jelasnya tujuan peruntukan pinjaman, adanya benda jaminan atau agunan, dan lain-lain (Iswi Hariyani; 2010: 11).

Fungsi Pinjaman/kredit bagi masyarakat adalah (Iswi Hariyani; 2010: 11) :

1. Menjadi *motivator* dan *dinamisator* peningkatan kegiatan perdagangan dan perekonomian.
2. Memperluas lapangan kerja bagi masyarakat.
3. Memperalncar arus barang dan arus uang.
4. Meningkatkan produktivitas dana yang ada.
5. Meningkatkan daya guna barang.
6. Meningkatkan kegairahan berusaha masyarakat.
7. Memperbesar modal usaha perusahaan.
8. Mengubah cara berpikir atau cara bertindak masyarakat untuk lebih ekonomis.

Tujuan penyaluran pinjaman/kredit adalah (Iswi Hariyani; 2010: 12) :

1. Memperoleh pendapatan dari bunga pinjaman.
2. Memanfaatkan dan memproduktifkan dana-dana yang ada.
3. Melaksanakan kegiatan operasional bank.
4. Memenuhi permintaan pinjaman dari masyarakat.
5. Memperlancar lalu lintas pembayaran.
6. Meningkatkan pendapatan dan kesejahteraan masyarakat.

Jika dilihat dari aspek bahasa, dapat dipahami bahwa "*credit/pinjaman*" bukan saja berarti hutang, tetapi bentuk dari percaya. Percaya yang dimaksud adalah bahwa pihak pemberi pinjaman mempercayai pihak pemberi pinjaman.

Jadi perlu dipahami bahwa pinjaman merupakan bentuk interaksi berdasarkan *kepercayaan* (Akhmad dan Adi Haryadi; 2006: 3).

II.5. Metode Decision Tree

Pohon Keputusan (*Decision Tree*) merupakan metode klasifikasi dan prediksi yang sangat kuat dan terkenal. Metode pohon keputusan mengubah fakta yang sangat besar menjadi pohon keputusan yang merepresentasikan aturan. Aturan dengan mudah dapat dipahami dengan bahasa alami dan mereka juga dapat diekspresikan dalam bentuk bahasa basis data seperti *Structured Query Language* untuk mencari *record* pada kategori tertentu (Emha taufik Luthfi dan Kusrini ; 2009: 13).

Sebuah Pohon Keputusan (*Decision Tree)* juga berguna untuk mengeksplorasi data, menemukan hubungan tersembunyi antara sejumlah calon variabel input dengan sebuah variabel target (Emha taufik Luthfi dan Kusrini ; 2009: 13).

Pohon keputusan adalah suatu gambaran permodelan dari suatu persoalan yang terdiri dari serangkaian keputusan yang mengarah ke solusi (Yuanda Ismiraldi:2007).

Sebuah pohon keputusan mungkin dibangun dengan seksama secara manual atau dapat tumbuh secara otomatis dengan menerapkan salah satu atau beberapa algoritma pohon keputusan untuk memodelkan himpunan data yangbelum terklasifikasi.Variabel tujuan biasanya dikelompokkan dengan pasti dan model pohon keputusan lebih mengarah pada perhitungan probabilitas dari tiap-tiap *record* terhadap kategori-kategori tersebut atau atau untuk-

mengklasifikasi *record* dengan mengelompokkannya dalam satu kelas (Kusrini dan Emha Taufiq Lutfi; 2009: 14).

Karena pohon keputusan memadukan antara eksplorasi data dan pemodelan, dia sangat bagus sebagai langkah awal dalam proses pemodelan bahkan ketika dijadikan sebagai model akhir dari beberapa teknik lain ((Emha taufik Luthfi dan Kusrini ; 2009: 13).

Menurut Kusrini (2007: 39), ada dua pendekatan dalam pembangunan sistem pendukung keputusan, yaitu:

1. Membangun sistem pendukung keputusan yang dibuat dengan bahasa pemrograman sesuai keperluan organisasi. Strategi pengembanganya adalah menggunakan bahasa pemrograman generik/umum, seperti visual basic, delphi, pascal, dan lain-lain.
2. Menggunakan generator sistem pendukung keputusan. Generator sistem pendukung keputusan adalah sebuah sistem aplikasi yang mengeliminasi penulisan kode program saat merancang dan membangun sistem pendukung keputusan. salah satu contoh generator sistem pendukung keputusan adalah lembar kerja (spreadsheet) elektronik, seperti Excel dan Lotus 1-2-3.

Untuk mengembangkan suatu sistem informasi, diperlukan metodologi pengembangan sistem, yaitu menyusun suatu sistem yang baru untuk menggantikan sistem lama secara keseluruhan atau memperbaiki sistem yang telah ada (Kusrini; 2007: 39).

Arsitektur sistem pendukung keputusan bisa terdiri dari beberapa sub sistem, yaitu (Kusrini; 2007: 25):

1. Subsistem manajemen data.
2. Subsistem manajemen model.
3. Subsistem antarmuka pengguna
4. Subsistem manajemen berbasis pengetahuan yang mendukung semua subsistem lain atau bertindak langsung sebagai suatu komponen independen dan bersifat opsional.

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

UML(*Unified Modelling Language*) merupakan alat komunikasi yang konsisten dalam mensupport para pengembang sistem saat ini. *UML(Unified Modelling Language)* telah diaplikasikan dalam bidang investasi perbankan, lembaga kesehatan, departemen pertahanan, sistem terdistribusi, sistem pendukung alat kerja, retail, sales dan suplier (Prabowo Pudjo Widodo dan Herlawati; 2010: 7**)** .

UML(*Unified Modelling Language*) adalah sebuah "bahasa" yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. *UML*(*Unified Modelling Language*) menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan menggunakan *UML*(*Unified Modelling Language*) dapat dubuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun (Prastuti Sulistyorini; 2009: 23-24).

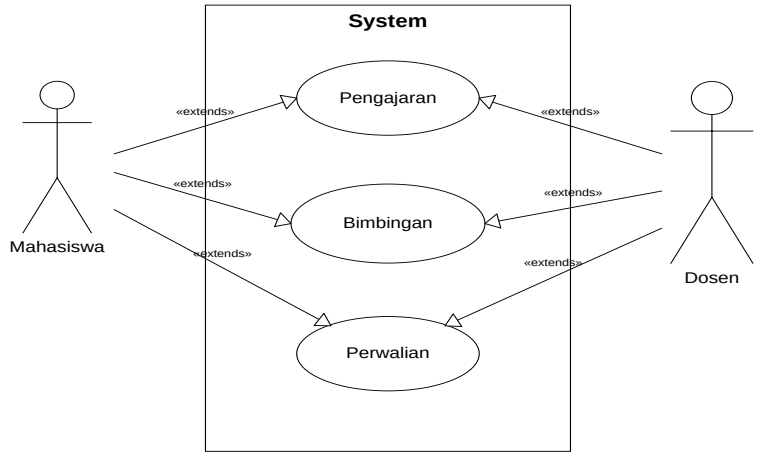
UML(*Unified Modelling Language*) merupakan sistem arsitektur yang bekerja dalam *object oriented analysis design* untuk menspesifikasikan, memvisualisasikan objek-objek dari sistem *software* untuk memodelkan bisnis dan komponenya (Dina Anggraini, dkk; 2008: 161).

II.6.1. Diagram *Use case*

Menurut Adi Nugroho (2010:35), *Use case* sesungguhnya merupakan unit koheren dari fungsionalitas sistem/perangkat lunak yang tampak dari luar dan diekspresikan sebagai urutan pesan-pesan yang dipertukarkan unit-unit sistem dengan satu atau lebih actor yang ada di luar sistem. Kegunaan *use case* sesungguhnya adalah untuk mendefinisikan suatu bagian perilaku sistem yang bersifat koheren tanpa perlu menyingkapkan struktur internal sistem/perangkat lunak yang sedang dikembangkan.

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 dari suatu sistem yang dibutuhkan serta diharapkan pengguna (Prastuti Sulistyorini; 2009: 24).

Notasi dan contoh diagram *use case* dapat dilihat pada gambar II.1 dibawah ini:



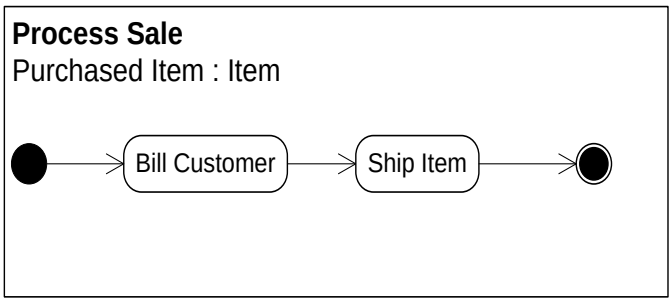
Gambar II.1 *Use Case* Diagram

Sumber: Adi Nugroho; 2010:34

II.6.2. Diagram *Activity*

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 (Adi Nugroho;2010:62). *Activity Diagram* memuat didalamnya activity state dimana activity state merepresentasikan eksekusi pernyataan dalam suatu prosedur atau kinerja suatu aktivitas dalam suatu aliran kerja (Adi Nugroho; 2010: 62).

Contoh diagram *Activity* dapat dilihat pada gambar II.2 dibawah ini:



Gambar II. 2. Activity Diagram

Sumber : Prabowo Pudjo Widodo dan Herlawati;2011:147

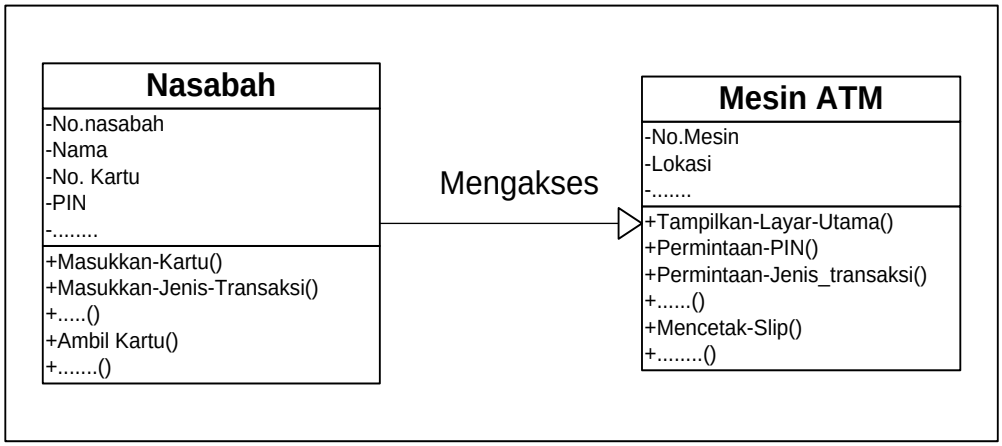
II.6.3. Diagram *Class*

Diagram kelas (*class diagram)* adalah inti dari proses pemodelan objek. Kemampuan meghasilkan kode program yang dimiliki oleh *class diagram* menyebabkan diagram ini memiliki hubungan yang khas dengan diagram *UML* lainya (Prabowo Pudjo Widodo dan Herlawati;2011:37-38).

Class sesungguhnya merepresentasikan suatu konsep diskret di dalam aplikasi yang dimodelkan, sesuatu yang bersifat fisik (misalnya mobil, pesawat terbang, dan sebagainya), sesuatu yang bersifat bisnis (misalnya pesanan), sesuatu yang bersifat logika (misalnya penjadwalan), sesuatu yang sangat terkait dengan aplikasi (misalnya tombol-tombol, ikon-ikon, dan sebagainya), sesuatu yang merupakan konsep yang dikenali dalam terminologi ilmu komputer (misalnya tabel *hash* atau berbagai metode pengurutan dan pencarian [*sorting* dan *searching*]), atau sesuatu yang bersifat perilaku (*behaviour*) (misalnya pekerjaan tertentu) (Adi Nugroho; 17).

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

Contoh diagram *class* dapat dilihat pada gambar II.3 dibawah ini:



Gambar II.3 *Class* Diagram

Sumber: Adi Nugroho, 2009:39

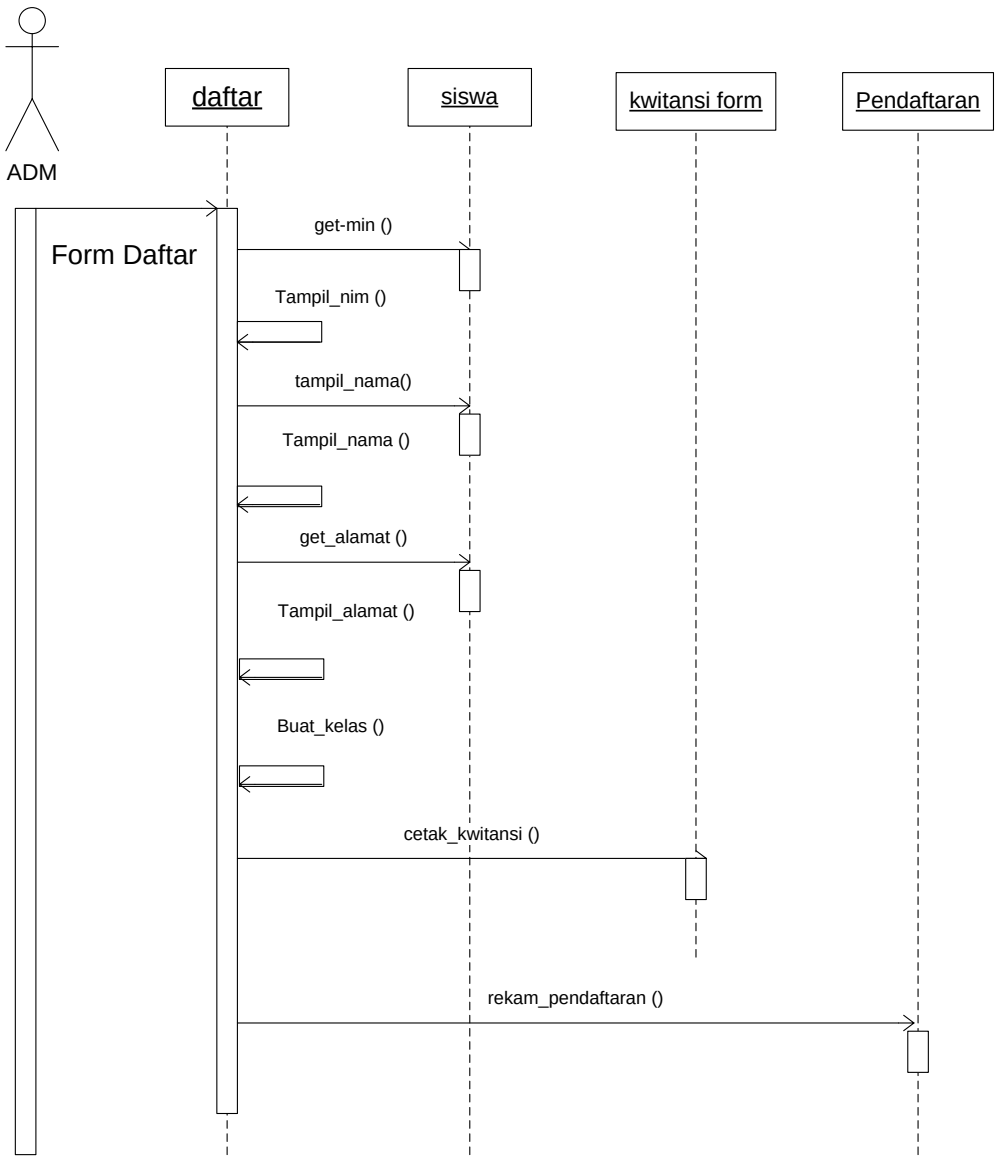
II.6.4. Diagram *Sequence*

Sequence Diagram atau yang disebut juga diagram urutan adalah diagram interaksi yang menekankan pada pengiriman pesan (*message*) dalam suatu waktu tertentu (Prabowo Pudjo Widodo dan Herlawati; 2011: 11).

Diagram *Sequence* secara khusus berasosiasi dengan *use case*. *Sequence* diagram memperlihatkan tahap demi tahap apa yang seharusnya terjadi untuk menghasilkan sesuatu di dalam use case. Diagram *sequence* sebaiknya digunakan

diawal tahap desain atau analisis karena kesederhanaannya dan mudah untuk dimengerti (Prastuti Sulistyorini; 2009: 27).

Contoh diagram *sequence* dapat dilihat pada gambar II.4 dibawah ini:



Gambar II.4 *Sequence* Diagram

Sumber: Prabowo Pudjo Widodo dan Herlawati;2010:221

II.7. Basis Data (*Database*)

Data merupakan representasi dari fakta mengenai suatu objek atau kejadian. Data dinyatakan dengan nilai yang berbentuk angka, deretan karakter, atau simbol. Contoh fakta mengenai biodata mahasiswa yang meliputi nama, alamat, jenis kelamin, agama yang dianut, dan lain-lain (Kusrini;2007:3).

Database adalah kumpulan data yang saling terkait yang diorganisasi untuk memenuhi kebutuhan dan struktur sebuah organisasi serta bisa digunakan oleh lebih dari satu orang dan lebih dari satu aplikasi (Kusrini; 2007: 33).

Basis data adalah kumpulan data yang saling berelasi. Data sendiri merupakan fakta mengenai objek, orang, dan lain-lain.data dinyatakan dengan nilai (angka, deretan karakter, atau *symbol*) (Kusrini; 2007: 2).

Tabel atau yang sebagian orang menyebutnya sebagai Relasi adalah komponen utama (komponen yang paling vital) dari suatu sistem basis data relasional yang berisi kolom dan baris tertentu di dalamnya (Adi Nugroho; 2010: 22).

Menurut Kusrini(2007:33-34), Ada 3(tiga) sumber data dalam sistem pendukung keputusan, yaitu:

1. Data internal

Data internal yang dimaksud adalah data yang sudah ada dalam suatu organisasi. Data tersebut bisa dikendalikan oleh organisasi tersebut. Data internal bisa berupa data mengenai orang, produk, layanan, dan proses-proses.

Contoh data internal adalah:

a. Data tentang pegawai

b. Data tentang peralatan dan mesin

c. Data penjualan

d. Data penjadwalan produksi

2. Data eksternal

Data eksternal adalah data yang tidak bisa dikendalikan oleh organisasi. Data tersebut berasal dari luar sistem. Contoh data eksternal adalah:

a. Peraturan perundangan

b. Harga pasar

c. Keadaan pesaing

d. Kurs Dolar

3. Data privat/personal

Data privat adalah data mengenai kepakaran/naluri dari *user* terhadap masalah yang akan diselesaikan. Dengan kata lain, data privat merupakan pendapat dari *user* mengenai variabel yang diperlukan dalam menyelesaikan masalah atau nilai dari suatu variabel.

Basis data dapat didefinisikan dalam berbagai sudut pandang seperti berikut (Kusrini; 2007: 2) :

1. Himpunan kelompok data yang saling berhubungan yang diorganisasi sedemikian rupa sehingga kelak dapat dimanfaatkan dengan cepat dan mudah.
2. Kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa tanpa pengulangan (*redundancy*) yang tidak perlu, untuk memenuhi kebutuhan.
3. Kumpulan tabel/*file*/arsip yang saling berhubungan yang disimpan dalam media penyimpanan elektronik.

Tujuan basis data adalah untuk mengatur data sehingga diperoleh kemudahan, ketepatan, dan kecepatan dalam pengambilan kembali (Kusrini; 2007: 2).

II.7.1 Normalisasi

Bentuk normal adalah keadaan tabel yang dihasilkan dengan menerapkan aturan sederhana berkaitan dengan konsep kebergantungan fungsional (*functional dependency*) pada relasi yang bersangkutan (Adi Nugroho; 2010: 34).

Tidak ada standar baku sampai sejauh mana kita perlu melaksanakan langkah normalisasi. Prinsipnya, lakukanlah normalisasi hingga anomali penyisipan, penambahan, dan penghapusan tidak ditemukan lagi (Adi Nugroho; 2010: 45-46).

Secara garis besar tahap normalisasi digambarkan sebagai berikut (Adi Nugroho; 2010: 34):

1. Bentuk Normal Pertama (1NF/*First Normal Form*). Bentuk normal pertama adalah suatu bentuk relasi dimana atribut bernilai banyak (*multivalues atribute*) telah dihilangkan sehingga kita akan menjumpai nilai tunggal pada perpotongan setiap baris dan kolom pada tabel.
2. Bentuk Normal Kedua (2NF/*Second Normal Form*). Semua kebergantungan fungsional (*functional dependency*) yang bersifat sebagian (*partial functional dependency*) telah dihilangkan).
3. Bentuk Normal Ketiga (3NF/*Third Normal Form*). Semua kebergantungan transitif *(transitive dependency*) telah dihilangkan.
4. *Boyce-Codd Normal Form (BCNF*/ *Boyce-Codd Normal Form*). Semua anomali yang tersisa dari hasil kebergantungan fungsional *(functional dependency)* di atas sudah dihilangkan.
5. Bentuk Normal Keempat (4NF/*Fourth Normal Form*). Semua anomali yang berasal dari kebergantungan banyak nilai (*multivalue dependency*) telah dihilangkan.

II.7.2 Kamus Data

Kamus data adalah suatu file yang terpisah yang menyimpan informasi seperti (Zulkifli Amsyah; 2005: 382):

1. Nama setiap *item*/jenis/kolom data.
2. Struktur data untuk tiap *item*.
3. Program yang mebggunakan tiap *item.*

4. Tingkat keamanan untuk setiap *item*.

Fungsi kamus data adalah mekanisme yang digunakan unutk menyediakan informasi mengenai beberapa *item* data yang bersangkutan (Zulkifli Amsyah; 2005: 382).

Kamus data berguna khusus bagi perlindungan timbulnya kelebihan data. Tanpa kamus data, pemakai dari lain bagian mungkin menyimpan versi identik dari *item* data yang sama pada beberapa lokasi, dimana masing-masing data mempunyai nama yang berbeda (Zulkifli Amsyah; 2005: 382).

Kamus data juga mengelola daftar "*password*" yang mengawasi akses ke sistem atau komputer (Zulkifli Amsyah; 2005: 382).

Kamus data mendiskripsikan setiap elemen data dalam basis data. Ini memungkinkan semua pengguna (dan programer) untuk berbagi pandangan yang sama mengenai sumber daya data, sehingga sangat memfasilitasi analisis kebutuhan pengguna (Hall Singleton; 2007: 148-149).

Kamus data (*data dictionary*) menyertakan informasi berupa user, hak istimewa (*privileges*) dan struktur internal *database*. Kamus data berisi metadata, yaitu informasi mengenai *database* (Ramon A.Mata Toledo dan Pauline K.Chusman; 2007: 3).

Contoh kamus data pada sebuah relasi antara *entity* dosen dengan mata kuliah lengkap (Yuhefizard; 2008: 33).

1. Dosen = (Nip, Nama Dosen, jurusan, bidang keahlian, alamat)
2. Matakuliah = (Kode Matakuliah, nama, sks, semester, pilihan)
3. Jadwal = (Nip, Kode Matakuliah, hari, jam, lokal)

Contoh kamus data pada sebuah relasi antara *entity* mahasiswa dengan dosen lengkap (Yuhefizard; 2008: 34):

1. Mahasiswa = (Nobp, Nama, tempat lahir, tanggal lahir, agama, jenis kelamin, PA, alamat)
2. Dosen = (Nip, Nama Dosen, jurusan, bidang keahlian, alamat)
3. Skripsi = (Nobp, Nip, Judul Skripsi, Tanggal Mulai, Tanggal Selesai)

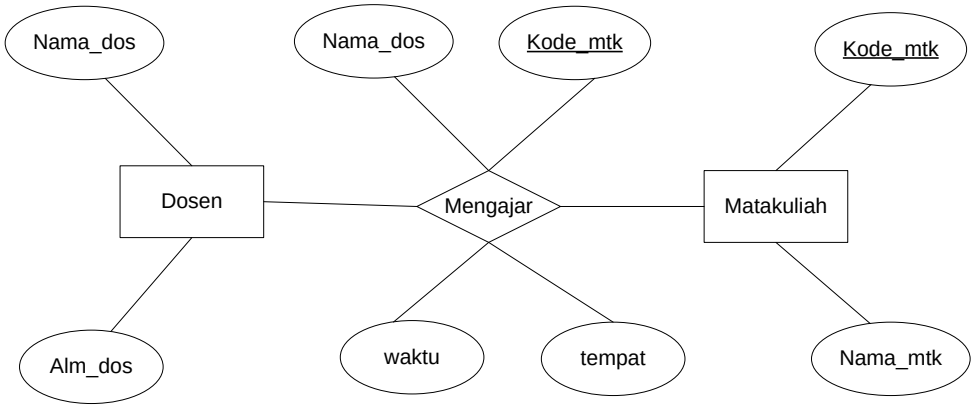
II.8. *Diagram Entity Relationship (Diagram E-R)*

Diagram E-R (*Entity Relationship*) digunakan untuk menggambarkan secara sistematis hubungan antar *entity-entity* yang ada dalam suatu sistem database menggunakan simbol-simbol sehingga lebih mudah dipahami (Yuhefizard; 2008: 17). Simbol-simbol yang boleh digunakan adalah:

1. Persegi Panjang, berfungsi untuk menyatakan suatu *entity*.
2. *Elips*, berfungsi untuk menyatakan suatu *attribute*, jika diberi garis bawah menandakan bahwa attribute tersebut merupakan *attribute*/*field* kunci.
3. Belah ketupat, menyatakan jenis relasi.

4. Garis, penghubung antara relasi dengan *entity* dan antara *entity* dengan *attribute*.

Misalnya hubungan antara *entity* Dosen dengan *entity* Matakuliah seperti terlihat pada diagram *E-R* dibawah ini:



Gambar II.5. *ERD(Entity Relationship Diagram*) hubungan *entity* Dosen dengan *entity* Matakuliah

Sumber: Yuhefizard; 2008: 17

Menurut Kusrini (2007: 21), Perancangan basis data dengan menggunakan model *E-R* adalah dengan menggunakan *Entity Realtionship Diagram(ERD)*. Sebuah *entity* adalah sebuah "benda" (*thing*) atau "objek"(*object*) di dunia nyata yang dapat dibedakan dari semua objek lainya. *Entity sets* adalah sekumpulan entity yang mempunyai tipe yang sama. Kesamaan tipe ini dapat dilihat dari atribut/*property* yang dimiliki oleh setiap *entity*. Misalnya:

1. kumpulan orang yang menyimpan uang pada suatu bank dapat didefinisikan sebagai *entity set* nasabah.

2. kumpulan orang yang belajar di perguruan tinggi didefinisikan sebagai mahasiswa.

ERD (Entity Relationship Diagram) menggambarkan berbagai relasi antar *entity* dalam suatu *database* (Sri Yulianto; 2008: 169).

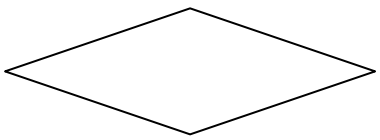
Terdapat tiga notasi dasar yang bekerja pada model diagram *E-R(Entity Relationship)* yang terdiri dari (Kusrini; 2007: 21):

1. *Entity Set* dilambangkan dengan bentuk persegi panjang, seperti tampak pada gambar berikut ini:

Gambar II.7. Lambang Entity Set

Sumber: Kusrini;2007:21

2. *Relationship* adalah hubungan diantara beberapa *entity*. *Relationship set* adalah sekumpulan relasi yang mempunyai tipe yang sama. *Relationship set* digambarkan dengan *diamond* seperti tampak pada gambar dibawah ini:



Gambar II.8. Lambang Relationship Set

Sumber: Kusrini;2007:21

3. Attributes

II.9. SQL Server 2005

SQL Server 2005 adalah *RDBMS (Relational database management system)* yang dikembangkan oleh microsoft. *SQL server* dapat digunakan sebagai basis- data untuk kebutuhan personal ataupun untuk organisasi. *SQL server* mempunyai beberapa versi, antara lain *Express Edition*, *Personal Edition*, dan *Enterprise Edition* (Alexander F.K Sibero;2010:10).

Sistem basis data relasional *(RDBMS-Relational Database Mnagement System)* adalah sistem penyimpanan data berbasis komputer yang elemen pokoknya adalah tabel-tabel yang saling berhubungan satu sama lain, dengan masing-masing tabel memiliki kolom-kolom (atribut-atribut) (atau sering juga disebut sebagai *field*) yang digunakan sebagai tempat untuk menyimpan data/ informasi yang diperlukan oleh aplikasi(baca:program komputer) yang kelak akan kita kembangkan (Adi Nugroho; 2010: 17-18).

SQL(Structure Query Language) adalah suatu suatu bahas standar yang digunakan oleh aplikasi untuk berkomunikasi dengan database server. Struktur *SQL (Structure Query Language)* terdiri dari dua bagian utama, yaitu *DDL(Data Definition Language*) dan *DML(Data Manipulation Language)*. *DDL(Data Definition Language*) adalah standar bahasa yanng digunakan untuk membuat definisi pada database, sedangkan *DML(Data Manipulation Language)* adalah standar bahasa yang digunakan unutk membuat formula pengolahan data. Selain *DDL(Data Definition Language*) dan *DML(Data Manipulation Language)*, pada struktur *SQL* juga terdapat istilah *Query*. *Query* adalah suatu kumpulan perintah *DDL(Data Definition Language*) ataupun *DML(Data Manipulation Language)*-

yang diformulasikan unutk memberikan hasil data ataupun perintah yang diharapkan (Alexander F.K Sibero; 2010:44).

II.10. Visual Basic.NET

Visual Basic.NET adalah bahasa pemroraman yang dikembangkan oleh perusahaan *microsoft*. *Visual Basic.NET* merupakan pengembangan dari versi sebelumnya, yaitu *Visual Basic 6.0*, yang memiliki karakteristik mudah untuk dipahami, namun andal dalam mengikuti tren teknologi perangkat lunak. Perbedaan mendasar antara *Visual Basic.NET* dengan versi-versi sebelumnya adalah kemampuan *OOP (Object Oriented Programming)* yang telah ditanamkan pada *Visual Basic.NET*. Saat ini *Visual Basic.NET* telah dikolaborasikan dengan-beberapa jenis aplikasi, seperti aplikasi *desktop* dan aplikasi berbasis *web* (Alexander F.K. Sibero;2010: 9).

Microsoft .NET Framework adalah sebuah *platform* baru dan revolusioner yang diciptakan oleh *Microsoft* untuk pengembangan *software*, tidak disebutkan "pengembangan *software* pada sistem operasi *windows*". Salah satu ide dibalik teknologi ini adalah keinginan mengintegrasikan berbagai sistem operasi yang berbeda-beda (Feri Djuandi; 2006: 1)

Visual Basic yang disingkat dengan *VB* merupakan bahasa pemrograman yang populer di kalangan programmer karena kemudahan pemakaian dan juga memiliki fitur-fitur yang sangat handal dalam mengembangkan aplikasi. *Visual Basic 2008* merupakan kelanjutan dari versi sebelumnya, yaitu *Visual Basic 2005*.

Versi 2008 ini sudah mengadopsi *dotnet framework 3.5*. Dalam *Framework* ini, beberapa pemrograman telah disempurnakan untul memudahkan dan juga-memberikan hasil pada aplikasi dengan sempurna (Cybertron Solution dan SmithDev Community; 2010: 1).

Aplikasi *Visual Basic 2008* dipaketkan pada aplikasi *Microsoft Visual Studio 2008.* Saat instalasi kita bisa memilih bahasa pemrograman yang akan diinstal pada komputer kita. Aplikasi yang ada pada *Visual Studio 2008* diantaranya adalah *VB, C#, dan C++* (Cybertron Solution dan SmithDev Community; 2010: 1).