BAB II

TINJAUAN PUSTAKA

II.1. Sistem Pendukung Keputusan (SPK)

Menurut Keen dan Scoot Morton Sistem Pendukung Keputusan merupakan penggabungan sumber - sumber kecerdasan individu dengan kemampuan komponen untuk memperbaiki kualitas keputusan. Sistem Pendukung Keputusan juga merupakan sistem informasi berbasis komputer untuk manajemen pengambilan keputusan yang menangani masalah - masalah semi struktur. (Pelita Informatika Budi Darma ; Nanda Abdurrahman Wahid ; 2014 : 93).

Sistem pendukung keputusan atau *Decision Support System* (DSS) adalah sistem informasi interaktif yang menyediakan informasi, pemodelan, dan pemanipulasian data yang digunakan untuk membantu pengambilan keputusan pada situasi yang semi terstruktur dan situasi yang tidak terstruktur dimana tak seorang pun tahu secara pasti bagaimana keputusan seharusnya dibuat. Konsep DSS dikemukakan pertama kali oleh Scoot Morton pada tahun 1971. (Pelita Informatika Budi Darma ; Nanda Abdurrahman Wahid ; 2014 : 93).

II.1.1. Karakteristik Sistem Pendukung Keputusan

Karakteristik sistem pendukung keputusan yaitu :

a. Mendukung proses pengambilan keputusan suatu organisasi atau perusahaan.

b. Adanya *interface* manusia/ mesin dimana manusia (user) tetap memegang kontrol proses pengambilan keputusan.
c. Mendukung pengambilan keputusan untuk membahas masalah terstruktur, semi terstruktur serta mendukung beberapa keputusan yang saling berinteraksi.
d. Memiliki kapasitas dialog untuk memperoleh informasi sesuai dengan kebutuhan.
e. Memiliki sub sistem yang terintegrasi sedemikian rupa sehingga dapat berfungsi sebagai kesatuan sistem.

II.1.2. Ciri-Ciri Sistem Pendukung Keputusan

Kriteria atau ciri-ciri sistem pendukung keputusan adalah sebagai berikut :

a. Banyak pilihan/alternatif.
b. Ada kendala atau surat.
c. Mengikuti suatu pola atau model tingkah laku, baik yang terstruktur maupun tidak terstruktur.
d. Banyak input/variabel.
e. Ada faktor resiko, dibutuhkan kecepatan, ketepatan, dan keakuratan (Konsep Data Mining vs Sistem Pendukung Keputusan ; Dicky Nofriansyah S.kom, M.kom ; 2014 : 2).

II.1.3. Fase Dalam Sistem Pendukung Keputusan

Tiga fase dalam pengambilan keputusan yaitu :

a. *Inteligence*

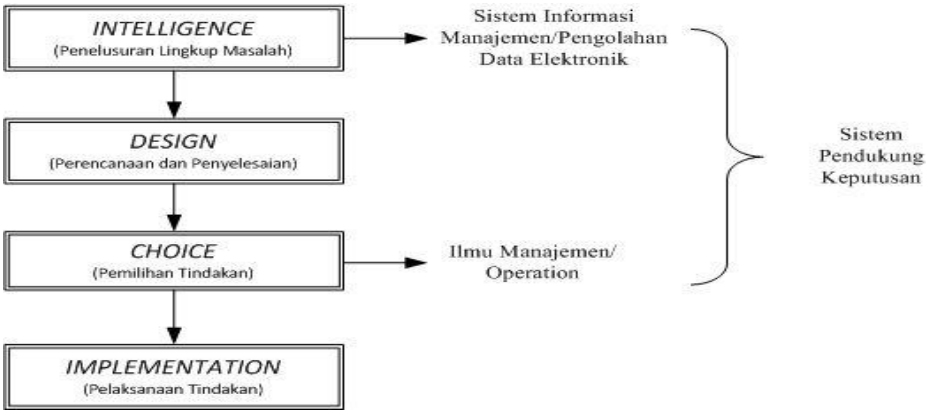
Tahap ini merupakan proses penelusuran dan pendeteksian dari ruang lingkup problematika secara proses pengenalan masalah. Data masukan diperoleh dan diuji dalam rangka mengidentifikasi masalah.

b. *Design*

Tahap ini merupakan proses menemukan, mengembangkan dan menganalisis alternatif tindakan yang bisa dilakukan. Tahap melakukan pengujian kelayakan solusi.

c. *Choice*

Pada tahap ini dilakukan proses pemilihan diantara berbagai alternatif tindakan yang mungkin dijalankan. Hasil pemilihan tersebut kemudian diimplentasikan kedalam proses pengambilan keputusan (Konsep Data Mining vs Sistem Pendukung Keputusan ; Dicky Nofriansyah S.kom, M.kom ; 2014 : 2-3).



Gambar II.1. Fase Proses Pengambilan Keputusan
(Sumber : Dicky Nofriansyah S.kom, M.kom ; 2014 : 2-3)

II.1.4. Komponen Sistem Pendukung Keputusan

Secara garis besar sistem pendukung keputusan dibangun oleh tiga komponen utama yaitu :

a. Sub sistem (*Database*)

   Sub sistem data merupakan komponen sistem pendukung keputusan yang berguna sebagai penyedia data bagi sistem. Data tersebut disimpan untuk diorganisasikan dalam sebuah basis data yang diorganisasikan oleh suat sistem yang disebut dengan sistem manajemen basis data (*database management system*).

b. Sub sistem model

   Model adalah suatu tiruan dari alam nyata. Kendala yang sering dihadapi dalam merancang model adalah bahwa model yang dirancang tidak mampu mencerminkan seluruh variabel alam nyata, sehingga keputusan yang diambil tidak sesuai dengan kebutuhan. Oleh karena itu, dalam menyimpan berbagai model harus diperhatikan dan harus dijaga fleksibelitasnya. Hal lain yang harus diperhatikan adalah pada setiap model yang disimpan hendaknya ditambahkan rincian, keterangan dan penjelasan yang komprehensif mengenai model yang dibuat.

c. Sub sistem dialog (*User Sistem Interface*)

   sub sistem dialog adalah fasilitas yang mampu mengintegrasikan sistem yang terpasang dengan pengguna secara interaktif, yang dikenal dengan sub sistem dialog. Melalui sub sistem dialog sistem diimplementasikan sehingga pengguna dapat berkomunikasi dengan sistem yang dibuat

(Konsep Data Mining vs Sistem Pendukung Keputusan ; Dicky Nofriansyah S.kom, M.kom ; 2014 : 3-4).

II.2. *Promethee*

Promethee (*Preference Ranking Organization Method for Enrichment Evaluation*) adalah satu dari beberapa metode penentuan urutan atau prioritas dalam analisis multikriteria. Metode ini dikenal sebagai metode yang efisien dan simple ,tetapi juga yang mudah diterapkan dibanding dengan metode lain untuk menuntaskan masalah multikriteria. Metode ini mampu mengakomodir kriteria

pemilihan yang bersifat kuantitatif dan kualitatif. Masalah utamanya adalah kesederhanaan, kejelasan dan kestabilan. Dugaan dari dominasi kriteria yang digunakan dalam promethee adalah penggunaan nilai dalam hubungan *outranking* (Jurnal Pelita Informatika Budi Dharma ; Nanda Abdurrahman Wahid ; 2014 :93).

Metode *Promethee* (*Preference Ranking Organizational Methods For Enrichment Evaluations*) adalah metode *outranking* yang menawarkan cara yang fleksibel dan sederhana kepada *user* untuk menganalisis masalah-masalah multikriteria (Jurnal Sarjana Teknik Informatika ; Cindra Onggo ; 2013 : 142).

II.2.1. Fungsi Preferensi Dalam *Promethee*

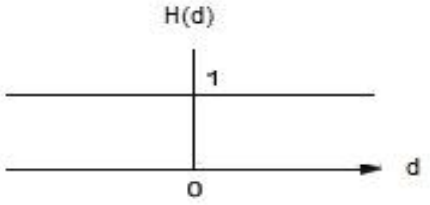
Dalam *promethee* disajikan enam bentuk fungsi preferensi kriteria. Hal ini tentu saja tidak mutlak, tetapi bentuk ini cukup baik untuk beberapa kasus. Untuk memberikan gambaran yang lebih baik terhadap area yang tidak sama, digunakan fungsi selisih nilai kriteria antara alternative H*(d)* dimana hal ini mempunyai

hubungan langsung pada fungsi preferensi P (Jurnal Telematika ; Bambang Yuwono dkk ; 2011 ; 64). Tipe preferensi yang digunakan dalam promethee adalah sebagai berikut :

1. Kriteria Biasa

$$H(d) \begin{cases} 0 \text{ jika } d = 0 \\ 1 \text{ jika } d \neq 0 \end{cases} \qquad (1)$$

dimana d = selisih nilai kriteria {d = f(a) – f(b)} Pada kasus ini tidak ada beda (sama penting) antara a dan b jika hanya jika *f(a) = f(b)*. Apabila nilai kriteria pada masing-masing alternatif memiliki nilai berbeda, pembuat keputusan membuat preferensi mutlak untuk alternatif yang memiliki nilai lebih baik (Jurnal Sarjana Teknik Informatika ; Cindra Onggo ; 2013 : 142).



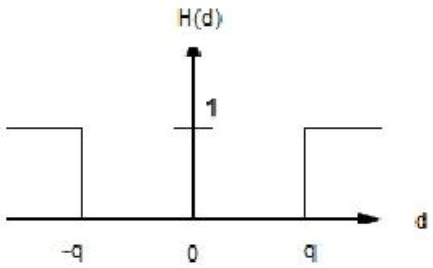
Gambar II.2. Kriteria Biasa

(Sumber : Cindra Onggo ; 2013 : 142)

2. Kriteria *Quasi*

$$H(d) \begin{cases} 0 \text{ jika } -q \leq d \leq q \\ 1 \text{ jika } d < -q \text{ atau } d > q \end{cases} \qquad (2)$$

Alternatif memiliki preferensi yang sama penting jika selisih dari masing-masing alternatif untuk kriteria tertentu tidak melebihi nilai q. Apabila selisih masing-masing alternatif melebihi nilai q, maka bentuk preferensi mutlak (Jurnal Sarjana Informatika ; Cindra Onggo ; 2013 : 142).



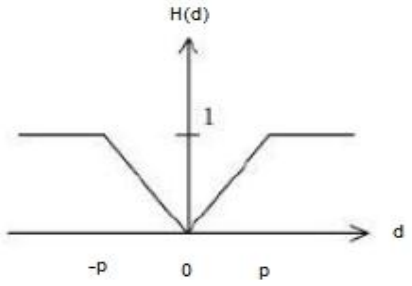
Gambar II.3. Kriteria *Quasi*

(Sumber : Cindra Onggo ; 2013 :142)

3. Kriteria dengan Preferensi Linier

$$H(d) \begin{cases} 1 \text{ jika } d < -p \text{ atau } d > p \\ d / p \text{ jika } -p \leq d \leq p \end{cases} \quad (3)$$

Selama nilai selisih memiliki nilai ebih rendah dari p, maka preferensi dari pembuat keputusan meningkat secara linier dengan nilai d. Jika nilai d lebih besar dibandingkan dengan nilai p, maka terjadi preferensi mutlak (Jurnal Sarjana Informatika ; Cindra Onggo ; 2013 : 143).



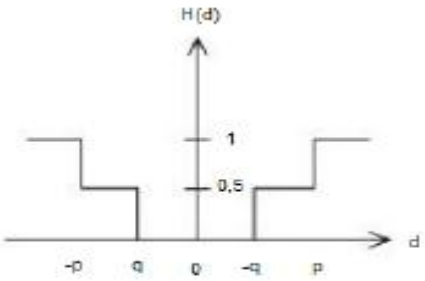
Gambar II.4. Kriteria Preferensi Linier

(Sumber : Cindra Onggo ; 2013 : 143)

4. Kriteria Level

$$H(d) \begin{cases} 0 \text{ jika } |d| \leq q \\ = 0{,}5 \text{ jika } q < |d| \leq p \\ 1 \text{ jika } p < |d| \end{cases} \qquad (4)$$

Jika berada di antara nilai q dan p, berarti H (d) = 0,5 (Jurnal Sarjana Informatika ; Cindra Onggo ; 2013 : 143).



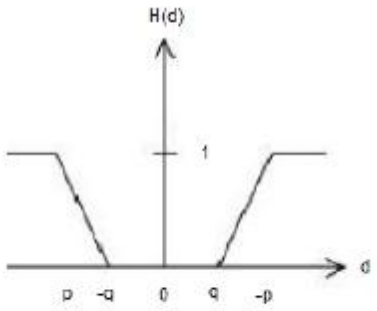
Gambar II.5. Kriteria Level

(Sumber : Cindra Onggo ; 2013 : 143)

5. Kriteria dengan preferensi linier dan area yang tidak berbeda

$$H(d) \begin{cases} 0 \text{ jika } |d| \leq q \\ = (|d| - q) / (p - q) \text{ jika } q < |d| \leq p \\ 1 \text{ jika } p < |d \end{cases} \qquad (5)$$

Pengambil keputusan mempertimbangkan peningkatan preferensi secara linier dan tidak berbeda, hingga preferensi mutlak dalam area antara dua kecenderungan q dan p (Jurnal Sarjana Informatika ; Cindra Onggo ; 2013 : 144).



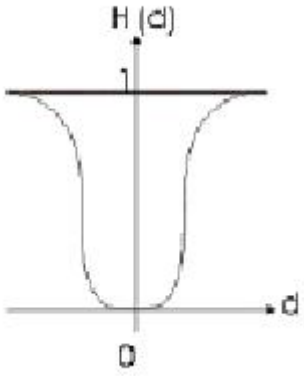
Gambar II.6. Kriteria dengan Area Preferensi Linier Yang Tidak Berbeda

(Sumber : Cindra Onggo ; 2013 : 144)

6. Kriteria *Gaussian*

$$H(d) = 1 - \exp\{ -d2 / 2\sigma 2 \} \qquad (6)$$

Fungsi ini bersyarat apabila telah ditentukan nilai σ (Jurnal Sarjana Informatika ; Cindra Onggo ; 2013 : 144).



Gambar II.7. Kriteria *Gaussian*

(Sumber : Cindra Onggo ; 2013 : 144)

Arah dalam grafik nilai *outranking* untuk setiap node a dalam grafik nilai *outranking* ditentukan berdasarkan *leaving flow*, dengan persamaan:

$$\phi^{+}(a) = \frac{1}{n-1} \sum_{x \in A} \wp\,(a,x) \qquad (7)$$

$\wp\,(a,x)$

Dimana menunujukan *preferens*i bahwa alternatif a lebih baik dari alternatif x. *Leaving Flow* adalah jumlah dari nilai garis lengkung yang memiliki arah menjauh dari node a dan hal ini merupakan karakter pengukuran *outranking*. Selain itu, juga merupakan suatu ukuran atau nilai yang menunjukkan kekuatan dari alternatif. *Entering Flow* merupakan suatu ukuran atau nilai yang menunjukan kelemahan dari alternatif. Secara simetris dapat ditentukan *entering flow* dengan persamaan :

$$\phi^{-}(a) = \frac{1}{n-1} \sum_{x \in A} \wp\ (x,a) \qquad (8)$$

Net flow menunjukkan suatu nilai total dari kekuatan dan kelemahan yang dimiliki oleh alternatif dalam penentuannya menggunakan persamaan:

$$\phi(a) = \phi^{+}(a) - \phi^{-}(a) \qquad (9)$$

II.3. PHP

PHP adalah kode atau skrip yang akan dieksekusi pada *server side*. Skrip PHP akan membuat suatu aplikasi daat diintegrasikan ke dalam HTML, sehingga suatu halaman web tidak lagi bersifat statis, namun menjadi bersifat dinamis. Sifat *server-side* berarti pengerjaan skrip dilakukan di *server* baru kemudian hasilnya dikirimkan ke *browser* (Deni Sutaji : 2012 ; 2).

Halaman web biasanya disusun dari kode-kode html yang disimpan dalam sebuah file berekstensi .html. File html ini dikirimkan oleh server (atau file) ke browser, kemudian browser menerjemahkan kode-kode tersebut sehingga menghasilkan suatu tampilan yang indah. Lain halnya dengan program php, program ini harus diterjemahkan oleh web-server sehingga menghasilkan kode html yang dikirim ke browser agar dapat ditampilkan. Program ini dapat berdiri sendiri ataupun disisipkan di antara kode-kode html sehingga dapat langsung ditampilkan bersama dengan kode-kode html tersebut. Program php dapat ditambahkan dengan mengapit program tersebut di antara tanda <? dan ?>. Tanda-tanda tersebut biasanya disebut tanda untuk escaping (kabur) dari kode html. File

html yang telah dibubuhi program php harus diganti ekstensi-nya menjadi .php3 atau .php.

PHP merupakan bahasa pemograman web yang bersifat server-side HTML = *embedded scripting*, di mana *script*-nya menyatu dengan HTML dan berada si server. Artinya adalah sintaks dan perintah-perintah yang pengguna berikan akan sepenuhnya dijalankan di *server* tetapi disertakan HTML biasa. PHP dikenal sebgai bahasa *scripting* yang menyatu dengan tag HTML, dieksekusi di *server* dan digunakan untuk membuat halaman web yang dinamis seperti ASP (*Active Server Pages*) dan JSP (*Java Server Pages*) (Rizka Prathesa ; 2012 : 4).

II.4. Basis Data

Secara umum untuk menjelaskan tentang pengertian basis data dapat ditinjau dari dua sisi, secara kharfiah dan istilah. Menurut pengertian secara kharfiah, basis data berasal dari dua kata yaitu basis dan data. Basis dapat diartikan sebagai suatu markas atau gudang, tempat bersarang atau tempat berkumpul. Data dapat diartikan sebagai representasi dari fakta dunia yang mewakili suatu objek (manusia, barang, peristiwa, keadaan, dsb) yang direkam dalam bentuk angka, huruf, simbol, teks, gambar, bunyi, atau kombinasinya. Adapun pengertian menurut istilah terdapat beberapa defenisi sebagai berikut :

1. Himpunan kelompok data (arsip) yang saling berhubungan yang diorganisasi sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah.

2. Kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa perulangan (*redudancy*) yang tidak perlu, untuk memenuhi berbagai kebutuhan.
3. Kumpulan *file*/tabel/arsip yang saling berhubungan yang disimpan dalam media penyimpanan tertentu.
4. Kumpulan data, yang dapat digambarkan sebagai aktifitas dari satu atau lebih organisasi yang berelasi.

Menurut Elmasari, penggunaan istilah basis data lebih dibatasi pada arti implisit yang khusus mempunyai beberapa pengertian yaitu :

1. Basis data merupakan penyajian suatu aspek dari dunia nyata (*real word* atau *miniworld*). Misalnya basis data perbankan, perpustakaan, pertanahan, perpajakan.
2. Basis data merupakan kumpulan data dari berbagai sumber yang secara logika mempunyai arti implisit. Sehingga apabila data terkumpul secara acak tanpa mempunyai arti, tidak dapat disebut basis data.
3. Basis data perlu dirancang, dibangun, dan data dikumpulkan untuk suatu tujuan tertentu.
4. Basis data dapat digunakan oleh beberapa pemakai dan beberapa aplikasi yang sesuai dengan kepentingan pemakai (Basis Data : Abdul Munif ; 2013 : 7-8).

II.4.1. Komponen Basis Data

Basis data merupakan suatu sistem yang dibangun oleh beberapa komponen diantaranya ada enam komponen pokok antara lain :

a. Perangkat keras (*Hardware*) dalam sistem komputer. Dalam sistem pengolahan basis data digital perangkat utama sebagai pengolah data adalah komputer.
b. Perangkat lunak aplikasi (*Software*) lain yang mendukung dan bersifat opsional. Perangkat lunak digunakan untuk mendukung proses pengelolaan basis data. Misalnya, bahasa pemograman c, *basic* pascal.
c. Sistem operasi (*Operating System*) merupakan perangkat lunak yang digunakan untuk mengelola aplikasi basis data dan penggunaan sumber daya komputer.
d. Basis data lain yang mempunyai keterkaitan dan hubungan dengan basis data itu sendiri. Berisi atau memiliki objek-objek basis data seperti, *file,* tabel, indeks. Mempunyai struktur baik untuk basis data maupun objek-objek secara detail.
e. Sistem Pengolahan Basis Data atau *Database Management System* (DBMS) merupakan program aplikasi untuk pengolahan basis data, seperti *microsoft access, oracle*, dan lain-lain.
f. Pemakai (*User*) yaitu pengguna yang terlibat dalam pengolahan basis dan penggunaan basis data (Basis Data ; Abdul Munif ; 2013 : 8).

II.4.2. Tujuan Penggunaan Basis Data

Beberapa tujuan penggunaan basis data adalah sebagai berikut :

a. Kecepatan dan kemudahan (*Speed*), melalui basis data diharapkan pengguna dapat melakukan penyimpanan, perubahan, dan menampilkan kembali dengan cepat dan mudah.
b. Efisiensi ruang penyimpanan (*Space*), penggunaan basis data mampu mengurangi pengulangan atau redudansi data. Hal ini dapat dilakukan dengan menerapkan sejumlah pengkodean atau dengan membuat relasi-relasi (dalam bentuk *file*) antara kelompok data yang saling berhubungan.
c. Keakuratan (*Accuracy*), melalui basis data keakuratan data lebih terjaga dengan menerapkan aturan dan batasan tertentu (*Constraint*), tipe data, domain data, dan keunikan data.
d. Ketersediaan (*Availability*), dengan basis data, data yang sudah tidak dipakai dapat dipisahkan dari sistem *database* yang sedang aktif. Hal ini dapat dilakukan dengan cara penghapusan atau memindahkannya ke media *backup*, untuk menghemat ruang penyimpanan. Selain itu dapat memanfaatkan teknologi jaringan komputer agar data yang berada di suatu lokasi atau cabang dapat juga diakses oleh lokasi atau cabang lainnya.
e. Kelengkapan (*Completeness*), agar data yang dikelola senantiasa lengkap baik relatif terhadap kebutuhan pemakai maupun terhadap waktu. Hal ini dapat dilakukan melalui penambahan *record-record*

data, perubahan struktur basis data, menambah *field* pada tabel, atau menambah tabel baru.

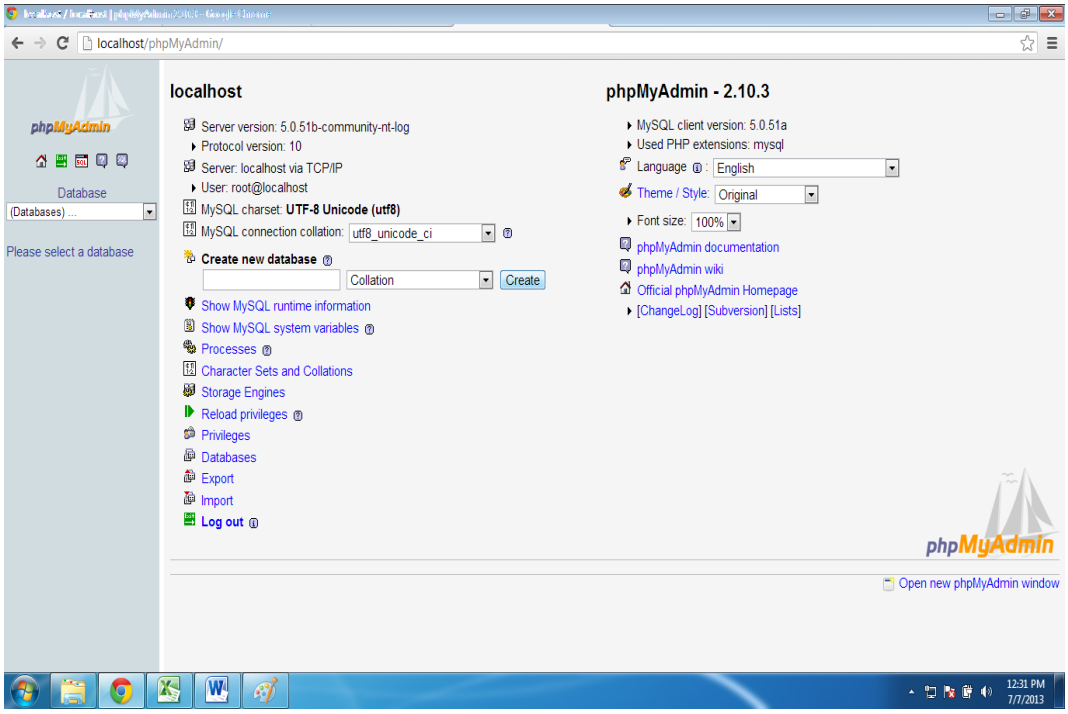
f. Keamanan (*Security*), walaupun tidak semua sistem basis data menerapkannya, keamanan dalam penggunaan basis data diperlakukan pada sistem yang besar dan serius. Dengan penerapan ini, setiap pengguna dibedakan hak aksesnya yakni, ditentukan objek-objek mana saja yang bisa diakses dan proses apa saja yang bisa dilakukan.

g. Kebersamaan (*Shareability*), agar data yang dikelola oleh sistem mendukung lingkungan *multiuser* (banyak pemakai) dengan menjaga atau menghindari munculnya *problem* baru seperti inkonsistensi data (karena terjadi perubahan data yang dilakukan oleh beberapa *user* dalam waktu yang bersamaan) atau kondisi *deadlock* (karena ada banyak pemakai yang saling menunggu untuk menggunakan data) (Basis Data ; Abdul Munif ; 2013 : 10).

II.5. MySQL

MySQL adalah suatu sistem manajemen basis data relasional (RDBMS-*Relational Database Management System*) yang mampu bekerja dengan cepat, kokoh, dan mudah digunakan. Contoh RDBMS lain adalah *Oracle, Sybase*. Basis data memungkinkan anda untuk menyimpan, menelusuri, menurutkan dan mengambil data secara efesien. *Server* MySQL yang akan membantu melakukan fungsionaliitas tersebut. Bahasa yang digunakan oleh MySQL tentu saja adalah SQL-*standar* bahasa basis data relasional di seluruh dunia saat ini.

MySQL dikembangkan, dipasarkan dan disokong oleh sebuah perusahaan Swedia bernama MySQL AB. RDBMS ini berada di bawah bendera GNU GPL sehingga termasuk produk *Open Source* dan sekaligus memiliki lisensi komersial. Apabila menggunakan MySQL sebagai basis data dalam suatu situs Web. Anda tidak perlu membayar, akan tetapi jika ingin membuat produk RDBMS baru dengan basis MySQL dan kemudian mengualnua, anda wajib bertemu mudah dengan lisensi komersial (Antonius Nugraha Widhi Pratama ; 2010 : 10).



Gambar II.2. Tampilan MySQL

(Sumber : Antonius Nugraha Widhi Pratama ; 2010 : 10)

II.6. Teknik Normalisasi

Normalisasi adalah teknik perancangan yang banyak digunakan sebagai pemandu dalam merancang basis data relasional. Pada dasarnya, normalisasi adalah proses dua langkah yang meletakkan data dalam bentuk tabulasi dengan

menghilangkan kelompok berulang lalu menghilangkan data yang terduplikasi dari tabel rasional.

Teori normalisasi didasarkan pada konsep bentuk normal. Sebuah tabel relasional dikatakan berada pada bentuk normal tertentu jika tabel memenuhi himpunan batasan tertentu. Ada lima bentuk normal yang tekah ditemukan.

II.6.1.Bentuk-bentuk Normalisasi

1. Bentuk normal tahap pertama (1" Normal Form)

Contoh yang kita gunakan di sini adalah sebuah perusahaan yang mendapatkan barang dari sejumlah pemasok. Masing-masing pemasok berada pada satu kota. Sebuah kota dapat mempunyai lebih dari satu pemasok dan masing-masing kota mempunyai kode status tersendiri.

ISBN	Thn_Terbit	ID_Pengarang	Nama_Pengarang	ID_Pengarang	Nama_Pengarang
12-1202-19222	1992	K0121	Aris M	K1021	Kosim P
11-1090-29101	2001	K1021	Kosim P		
11-1090-29102	2001	K2091	K Odelia	K0121	Aris M
12-1201-90871	2002	K2092	Renaldi	K2091	K Odelia
13-2089-12910	2001	K2019	Samsuri J		

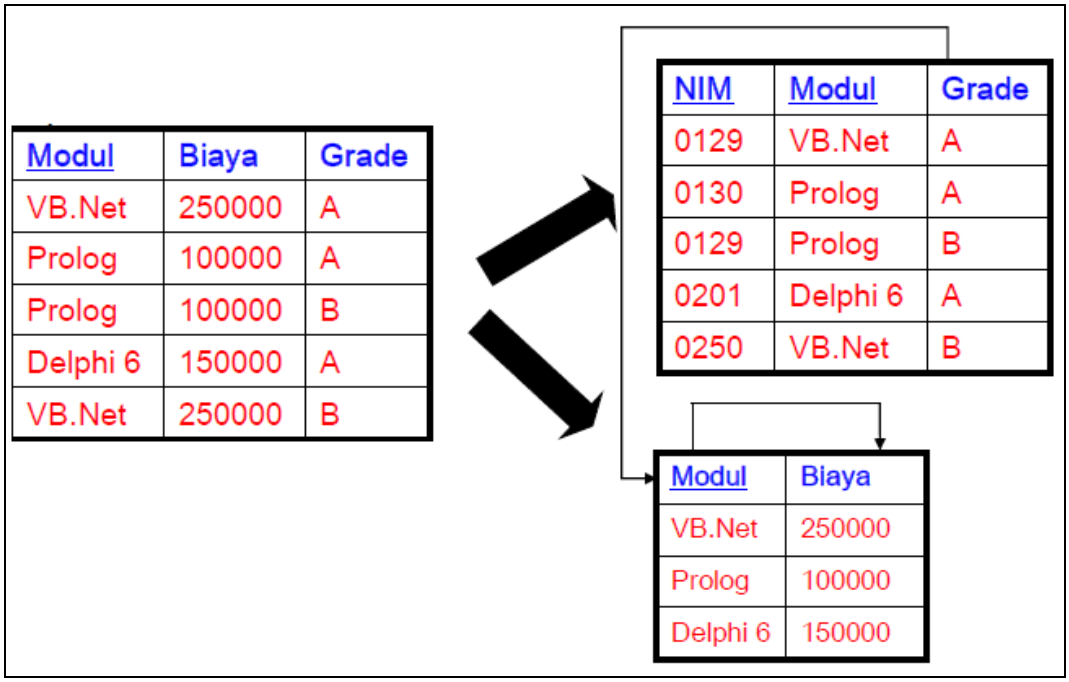
ISBN	Thn_Terbit	ID_Pengarang	Nama_Pengarang
12-1202-19222	1992	K0121	Aris M
12-1202-19222	1992	K1021	Kosim P
11-1090-29101	2001	K1021	Kosim P
11-1090-29102	2001	K2091	K Odelia
11-1090-29102	2001	K0121	Aris M
12-1201-90871	2002	K2092	Renaldi
12-1201-90871	2002	K2091	K Odelia
13-2089-12910	2001	K2019	Samsuri J

Gambar II.3. Normalisasi 1NF

Sumber : Janner Simarmata ; 2010 : 76

2. Bentuk normal tahap kedua (2nd normal form)

Definisi bentuk normal kedua menyatakan bahwa tabel dengan kunci utama gabungan hanya dapat berada pada 1NF, tetapi tidak pada 2NF. Sebuah tabel relasional berada pada bentuk normal kedua jika dia berada pada bentuk normal kedua jika dia berada pada 1NF dan setiap kolom bukan kunci yang sepenuhnya tergantung pada seluruh kolom yang membentuk kunci utama.



Gambar II.4. Normalisasi 2NF

Sumber : Janner Simarmata ; 2010 : 76

3. Bentuk normal tahap ketiga (3rd normal form)

Bentuk normal ketiga mengharuskan semua kolom pada tabel relasional tergantung hanya pada kunci utama. Secara definisi, sebuah tabel berada pada bentuk normal ketiga (3NF) jika tabel sudah berada pada 2NF dan setiap kolom yang bukan kunci tidak tergantung secara transitif pada kunci utamanya.

S	P	Qty
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P2	200
S4	P4	399
S4	P5	400

Gambar II.5. Normalisasi 3NF

Sumber : Janner Simarmata ; 2010 : 76

4. Boyce Code Normal Form (BCNF)

Setelah 3NF, semua masalah normalisasi hanya melibatkan tabel yang mempunyai tiga kolom atau lebih dan semua kolom adalah kunci. Banyak praktisi berpendapat bahwa menempatkan entitas pada 3NF sudah cukup karena sangat jarang entitas yang berada pada 3NF bukan merupakan 4NF dan 5NF.

Modul	Dosen
VB.Net	Ajib
Prolog	Aris
Prolog	Aris
VB Net	Budi
Prolog	Jono
VB.Net	Budi

NIM	Dosen
0129	Ajib
0130	Aris
0129	Aris
0201	Budi
0250	Jono
0260	Budi

Dosen	Modul
Ajib	VB.Net
Aris	Prolog
Jono	Prolog
Budi	VB.Net

Gambar II.6. Normalisasi BCNF

Sumber : Janner Simarmata ; 2010 : 76

5. Bentuk Normal Tahap Keempat dan Kelima

Sebuah tabel relasional berada pada bentuk normal keempat (4NF) jika dia dalam BCNF dan semua ketergantungan multivalue merupakan ketergantungan fungsional. Bentuk normal keempat (4NF) didasarkan pada konsep ketergantungan multivalue (MVD).

Sebuah tabel berada pada bentuk normal kelima (5NF) jika ia tidak dapat mempunyai dekomposisi lossless menjadi sejumlah tabel lebih kecil. Empat bentuk normal pertama berdasarkan pada konsep ketergantungan fungsional, sedangkan bentuk normal kelima berdasarkan pada konsep ketergantungan gabungan (*join dependence*) (Janner Simarmata ; 2010 : 76).

FirstName	LastName	Major	Level
Jack	Jones	LIS	Graduate
Lynn	Lee	LIS	Graduate
Mary	Ruiz	Pre-Medicine	Undergraduate
Lynn	Smith	Pre-Law	Undergraduate
Jane	Jones	LIS	Graduate

Gambar II.7. Normalisasi 4NF

Sumber : Janner Simarmata ; 2010 : 76

II.7. *Unified Modeling Language* (UML)

Menurut Windu Gata (2013 : 4) Hasil pemodelan pada OOAD terdokumentasikan dalam bentuk *Unified Modeling Language* (UML). UML adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak.

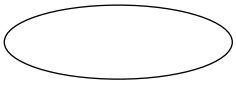
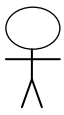
UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. UML saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem.

Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML adalah sebagai berikut :

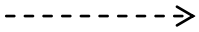
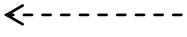
1. *Use case* Diagram

Use case diagram merupakan pemodelan untuk kelakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Dapat dikatakan *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Simbol-simbol yang digunakan dalam *use case* diagram, yaitu :

Tabel II.1. Simbol *Use Case*

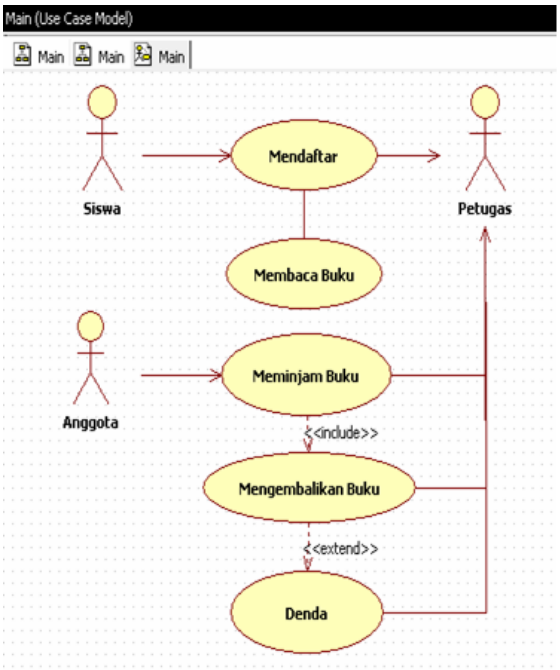
Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukan pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal nama *use case*.
	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan *use case*, tetapi tidak memiliki control terhadap *use case*.

Tabel II.1. Simbol *Use Case* (Lanjutan)

	Asosiasi antara aktor dan *use case*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengidikasikan aliran data.
	Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengidinkasikan bila aktor berinteraksi secara pasif dengan sistem.
	*Include*, merupakan di dalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain, contohnya adalah pemanggilan sebuah fungsi program.
	*Extend*, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

(Sumber : Windu Gata ; 2013 : 4)

Contoh dari pembuatan *use case diagram* dapat dilihat pada gambar II.8 berikut :



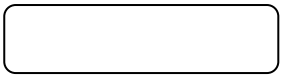
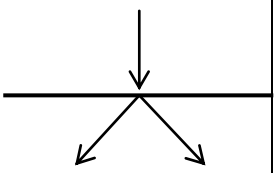
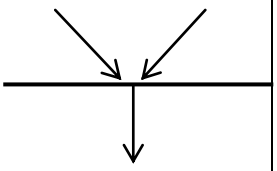
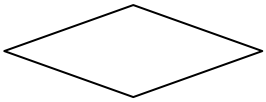
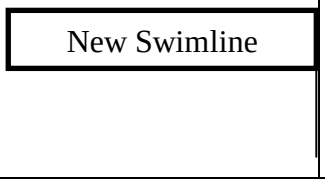
Gambar. II.8. *Use Case Diagram*

(Sumber : Windu Gata ; 2013 : 4)

2. Diagram Aktivitas (*Activity Diagram*)

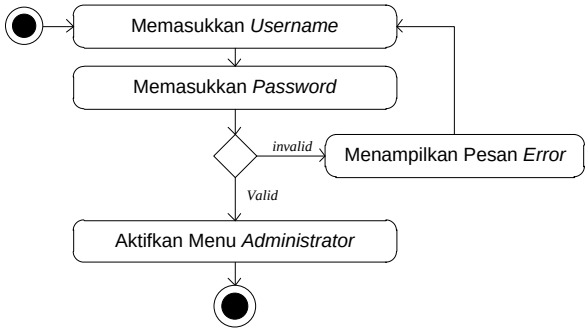
Activity Diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Simbol-simbol yang digunakan dalam *activity diagram,* yaitu :

Tabel II.2. Simbol *Activity Diagram*

Gambar	Keterangan
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activites*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu.
	*Join* (penggabungan) atau rake, digunakan untuk menunjukkan adanya dekomposisi.
	*Decision Points*, menggambarkan pilihan untuk pengambilan keputusan, *true*, *false.*
	*Swimlane*, pembagian *activity* diagram untuk menunjukkan siapa melakukan apa.

(Sumber : Windu Gata ; 2013 : 6)

Contoh dari pembuatan *activity diagram* dapat dilihat pada gambar II.9 berikut :



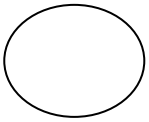
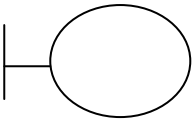
Gambar. II.9. *Activity Diagram*

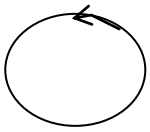
(Sumber : Windu Gata ; 2013 : 6)

3. Diagram Urutan (*Sequence Diagram*)

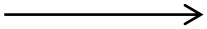
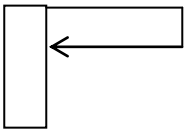
Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Simbol-simbol yang digunakan dalam *sequence diagram*, yaitu :

Tabel II.3. Simbol *Sequence Diagram*

Gambar	Keterangan
	*Entity Class*, merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.
	*Boundary Class*, berisi kumpulan kelas yang menjadi *interface* atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan formentry dan *form* cetak.

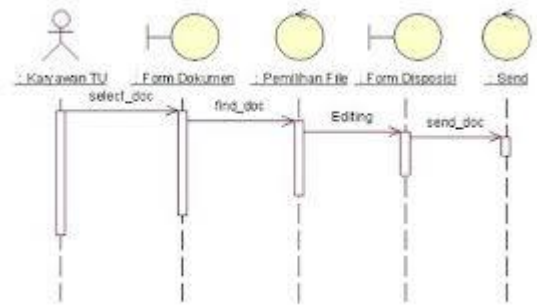
	*Control class,* suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek.

Tabel II.3. Simbol *Sequence Diagram* (Lanjutan)

	*Message*, simbol mengirim pesan antar *class*.
	*Recursive*, menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.
	*Activation, activation* mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.
	Lifeline, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*.

(Sumber : Windu Gata ; 2013 : 7)

Contoh dari pembuatan *sequence diagram* dapat dilihat pada gambar II.10 berikut :



Gambar. II.16. *Sequence Diagram*

(Sumber : Windu Gata ; 2013 : 7)

4. *Class Diagram* (Diagram Kelas)

Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggng jawab entitas yang menentukan perilaku sistem.

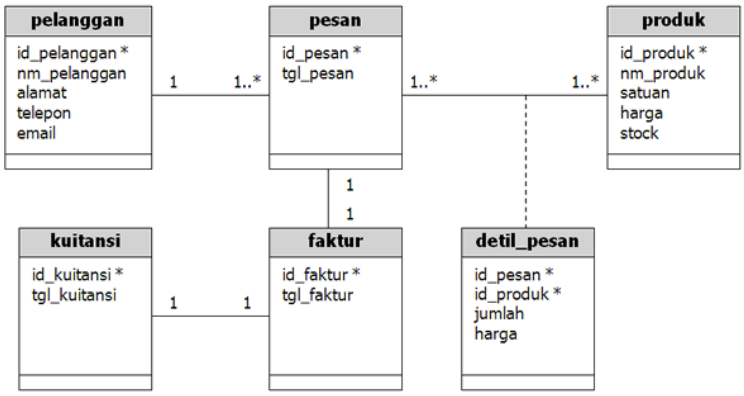
Class diagram juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan *constraint* yang berhubungan dengan objek yang dikoneksikan. *Class diagram* secara khas meliputi: Kelas (*Class*), Relasi, *Associations*, *Generalization* dan *Aggregation*, Atribut (*Attributes*), Operasi (*Operations/Method*), *Visibility*, tingkat akses objek eksternal kepada suatu operasi atau atribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *multiplicity* atau kardinaliti.

Tabel II.4. *Multiplicity Class Diagram*

Multiplicity	**Penjelasan**
1	Satu dan hanya satu
0..*	Boleh tidak ada atau 1 atau lebih
1..*	1 atau lebih
0..1	Boleh tidak ada, maksimal 1
n..n	Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimum 4

(Sumber : Windu Gata ; 2013 : 8)

Contoh dari pembuatan *Class diagram* dapat dilihat pada gambar II.11 berikut :



Gambar. II.11. *Class Diagram*

(Sumber : Windu Gata ; 2013 : 8)