BAB II

TINJAUAN PUSTAKA

II.1 Pengertian Sistem Pendukung Keputusan

Sistem Pendukung Keputusan atau *Decision Support System* yang selanjutnya kita singkat dalam skripsi ini menjadi SPK, secara umum didefenisikan sebagai sebuah sistem yang mampu memberikan kemampuan baik, kemampuan pemecahan masalah, maupun kemampuan pengkomunikasian untuk masalah semi-terstruktur. Secara khusus, SPK didefinisikan sebagai sebuah sistem yang mendukung kerja seorang manajer dalam memecahkan masalah semi-terstruktur dengan cara memberikan informasi ataupun usulan menuju pada keputusan tertentu.

Menurut Alter dalam Kusrini (2007, h. 15) Sistem Pendukung Keputusan atau *Decision Support Sistem* (DSS) "merupakan sistem informasi interaktif yang menyediakan informasi, pemodelan, dan pemanipulasian data. Dan sistem itu digunakan untuk membantu pengambilan keputusan dalam situasi yang semiterstruktur dan situasi yang tidak terstruktur, di mana tak seorang pun tahu secara pasti bagaimana keputusan seharusnya dibuat".

Tujuan dari sistem pendukung keputusan antara lain :

1. Membantu pegawai dalam pengambil keputusan atas masalah semiterstruktur.
2. Memberikan dukungan atas pertimbangan pegawai dan bukannya dimaksudkan untuk menggantikan fungsi pegawai.

3. Meningkatkan efektifitas keputusan yang diambil pegawai lebih dari pada perbaikan efisiensinya.
4. Meningkatkan kecepatan komputasi, untuk melakukan banyak komputasi secara cepat dengan biaya yang rendah.
5. Peningkatan produktifitas, produktifitas biasa ditingkatkan menggunakan peralatan optimalisasi yang menentukan cara terbaik untuk menjalankan sebuah bisnis.
6. Memberikan dukungan kualitas. Komputer bisa meningkatkan kualitas keputusan yang dibuat. Dengan komputer menilai berbagai pengaruh secara cepat dan ekonomis.
7. Meningkatkan daya saing. Teknologi pengambil keputusan bisa menciptakan pemberdayaan dengan cara memperbolehkan seorang untuk membuat keputusan yang baik dan tepat.
8. Mengatasi keterbatasan kognitif dalam pemrosesan dan penyimpanan

   Pembuatan keputusan merupakan fungsi utama seorang manajer atau administrator. Kegiatan pembuatan keputusan meliputi pengidentifikasian masalah, pencarian alternatif penyelesaian masalah, evaluasi dari alternatif-alternatif tersebut dan pemilihan alternatif keputusan yang baik. Kemampuan seorang manajer dalam membuat keputusan dapat ditingkatkan apabila ia mengetahui dan menguasai teori dan teknik pembuatan keputusan. Dengan peningkatan kemampuan manajer dalam pembuatan keputusan diharapkan dapat ditingkatkan kualitas keputusan yang dibuatnya, dan hal ini tentu akan meningkatkan efesiensi kerja manajer yang bersangkutan.

II.1.1. Keuntungan Sistem Pendukung Keputusan

Dengan berbagai karakter khusus yang dimiliki Sistem Pendukung Keputusan, SPK (Sistem Pendukung Keputusan) dapat memberikan berbagai manfaat dan keuntungan. Keuntungan yang dapat diambil dari SPK adalah:

1. Mampu mendukung pencarian solusi dari masalah yang kompleks.
2. Respon cepat pada situasi yang tak diharapkan dalam kondisi yang berubah-ubah.
3. Mampu untuk menerapkan berbagai strategi yang berbeda pada konfigurasi berbeda secara cepat dan tepat.
4. Pandangan dan pembelajaran baru.
5. Mempasilisasi komunikasi.
6. Meningkatkan control manajement dan kinerja.
7. Menghemat biaya.
8. Keputusan lebih cepat.
9. Meningkatkan produktifitas analisis.

II.2. Konsep dasar *Multi Attribute Decision Making (MADM)*

MADM menyelesaikan masalah-masalah dalam ruang diskret. Oleh karena itu, pada MADM biasanya digunakan untuk melakukan penilaian atau seleksi terhadap beberapa alternatif dalam jumlah yang terbatas.

Menurut Rudolphi dalam Kusumadewi dkk. (2006, h. 72), Pada dasarnya, proses MADM dilakukan melalui 3 tahap, yaitu penyusunan komponen-komponen situasi, analisis, dan sintesis informasi. Pada tahap penyusunan komponen,

komponen situasi, akan dibentuk tabel taksiran yang berisi identifikasi alternatif dan spesifikasi tujuan, kriteria dan atribut.

Multi Attribute Decision Making (MADM) adalah suatu metode yang digunakan untuk mencari alternatif otptimal dari sejumlah alternatif dengan kriteria tertentu. Tahap analisis dilakukan melalui 2 langkah. Pertama, mendatangkan taksiran dari besaran yang potensial, kemungkinan dan ketidakpastian yang berhubungan dengan dampak-dampak yang mungkin pada setiap alternatif. Kedua, meliputi pemilihan dari preferensi pengambil keputusan untuk setiap nilai, dan ketidak pedulian terhadap resiko yang timbul. Pada langkah pertama, beberapa metode menggunakan fungsi distribusi $|p_j(x)|$ yang menyatakan probabilitas kumpulan atribut $|a_k|$ terhadap setiap alternatif $|A_i|$. Konsekuen juga dapat ditentukan secara langsung dari agregasi sederhana yang dilakukan pada informasi terbaik yang tersedia. Demikian pula, ada beberapa cara untuk menentukan preferensi pengambil keputusan pada setiap konsekuen yang dapat dilakukan pada langkah kedua. Metode yang paling sederhana adalah untuk menurunkan bobot atribut dan kriteria adalah dengan fungsi utilitas atau penjumlahan terbobot.

Dalam Kusumadewi dkk. (2006, h. 72), Zimermann menyatakan bahwa, secara umum, model *multi attribute decision making* dapat didefinisikan sebagai berikut:

Misalkan $A = \{a_i \mid i = 1, \ldots, n\}$ adalah himpunan alternatif keputusan dan $C = \{c_j \mid j = 1, \ldots, m\}$ adalah himpunan tujuan yang diharapkan, maka akan ditentukan

alternatif x^0 yang memiliki derajat harapan tertinggi terhadap tujuan-tujuan yang relevan c_j.

Namun sebagian besar pendekatan MADM dilakukan melalui 2 langkah, yaitu : pertama, melakukan agregasi terhadap keputusan-keputusan yang tanggap terhadap semua tujuan pada setiap alternatif. Sedangkan yang kedua, melakukan perankingan alternatif-alternatif keputusan tersebut berdasarkan hasil agregasi keputusan.

Dengan demikian, bisa dikatakan bahwa, masalah *multi-attribute decision making* (MADM) adalah mengevaluasi m alternatif A_i (i=1,2,….,m) terhadap sekumpulan atribut atau kriteria C_j (j=1,2,….,n), dimana setiap atribut saling tidak bergantung satu dengan yang lainnya. Matriks keputusan setiap alternatif terhadap setiap atribut x, diberikan sebagai:

$$x = \begin{pmatrix} x_{11} & x_{12} & \ldots. & x_{1n} \\ x_{21} & x_{22} & \ldots. & x_{2n} \\ \ldots. & \ldots. & \ldots. & \ldots. \\ \ldots. & \ldots. & \ldots. & \ldots. \\ x_{m1} & x_{m2} & \ldots. & x_{mn} \end{pmatrix}$$

Dimana x_{ij} merupakan rating kinerja alternatif ke-i terhadap atribut ke-j. Nilai bobot yang menunjukkan tingkat kepentingan relatif setiap atribut, diberikan sebagai, w :

$w = \{ w_1, w_2, \ldots., w_n \}$

Rating kinerja (x), dan nilai bobot (w) merupakan nilai utama yang merepresentasikan preferensi absolut dari pengambil keputusan/ masalah MADM diakhiri dengan proses perangkingan untuk mendapatkan alternatif terbaik yang

diperoleh berdasarkan nilai keseluruhan preferensi yang diberikan (Yeh dalam Kusumadewi dkk. (2006, h. 73).

Beberapa metode yang dapat digunakan untuk menyelesaikan masalah MADM, antara lain sebagai berikut :

1. *Simple Additive Weighting Method* (SAW)
2. *Weighted Product Model* (WPM)
3. *ELECTRE*
4. *Technique for Order Preference by Similarity to Ideal Solution* (TOPSIS)
5. *Analytic Hierarchy Process* (AHP)

II.3. Metode *Profile Matching*

Profile Matching adalah sebuah mekanisme pengambilan keputusan dengan mengasumsikan bahwa terdapat tingkat variable prediktor yang ideal yang harus dimiliki oleh karyawan, bukannya tingkat minimal yang harus dipenuhi atau dilewati (Kusrini, 2010).

Secara garis besar merupakan proses membandingkan antara kompetensi individu kedalam kompetensi jabatan sehingga dapat diketahui perbedaan kompetensinya (disebut juga *gap*), semakin kecil *gap* yang dihasilkan maka bobot nilainya semakin besar yang berarti memiliki peluang lebih besar untuk karyawan menempati posisi tersebut (Asfan Muqtadir, Irwan Purdianto, 2013).

Metode *Profile Matching* merupakan proses membandingkan antara kompetensi jabatan sehingga dapat diketahui perbedaan kompetensinya (GAP), semakin kecil GAP yang dihasilkan maka bobot nilainya semakin besar yang

berarti memiliki peluang lebih besar untuk karyawan menempati posisi tersebut (Rahmat Hidayat, Nita Merlina, 2013).

Profile matching merupakan suatu proses yang sangat penting dalam manajemen SDM di mana terlebih dahulu ditentukan kompetensi (kemampuan) yang diperlukan oleh suatu jabatan (Andreas Handojo, Djoni H. Setiabudi, Rachma Yunita)

Langkah Penyelesaian :

Gap = Profil Karyawan – Profile Jabatan

Setelah menentukan bobot nilai *gap* untuk ketiga aspek yaitu aspek kapasitas intelektual, sikap kerja dan perilaku dengan cara yang sama. Kemudian tiap aspek dikelompokkan menjadi 2 (dua) kelompok yaitu kelompok *Core Faktor* dan *Secondary Faktor*. Untuk perhitungan *Core Faktor* dapat ditunjukkan pada rumus di bawah ini:

$$NCF = \frac{\Sigma NC(I, s, p)}{\Sigma IC} \quad \ldots\ldots (1)$$

Di mana :

NCF : Nilai rata-rata *core faktor*

NC(*i, s, p*) : Jumlah total nilai *core faktor (Intelektual, Sikap kerja, Perilaku)*

IC : Jumlah *item core faktor*

Sedangkan untuk perhitungan *secondary faktor* dapat ditunjukkan pada rumus di bawah ini:

$$NCS = \frac{\Sigma NS(I, s, p)}{\Sigma IS} \quad \ldots\ldots (2)$$

Di mana :

NSF : Nilai rata-rata *secondary faktor*

NS(*i, s, p*) : Jumlah total nilai *secondary faktor (Intelektual, Sikap kerja,Perilaku)*

IS : Jumlah *item secondary faktor*

Dari hasil perhitungan dari tiap aspek di atas kemudian dihitung nilai total berdasar presentasi dari *core* dan *secondary* yang diperkirakan berpengaruh terhadap kinerja tiap-tiap profil. Contoh perhitungan dapat dilihat pada rumus di bawah ini:

$$N(i, s, p) = (x)\%NCF(i, s, p) + (x)\%NSF(i, s, p) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3)$$

Di mana :

(i,s,p) : (Intelektual, Sikap Kerja, Perilaku)

N(i,s,p) : Nilai total dari aspek

NCF(i,s,p) : Nilai rata-rata *core faktor*

NSF(i,s,p) : Nilai rata-rata *secondary faktor*

(x)% : Nilai persen yang di*input*kan

Hasil akhir dari proses ini adalah ranking dari kandidat yang diajukan untuk mengisi suatu jabatan tertentu. Penentuan ranking mengacu pada hasil perhitungan tertentu. Perhitungan tersebut dapat ditunjukkan pada rumus di bawah ini:

$$Ha = (x)\%Ni + (x)\%Ns + (x)\%Np \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (4)$$

Di mana :

Ha : Hasil Akhir

Ni : Nilai Kapasitas Intelektual

Ns : Nilai Sikap Kerja

Np : Nilai Perilaku

(x)% : Nilai Persen yang di*input*kan

Contoh kasus :

Np SALES-0001 = (60% x 4,5) . (40% x 4,5) = 4,5

Np SALES-0001 = (60% x 4,75) . (40% x 4,5) = 4,65

Np SALES-0001 = (60% x 4) . (40% x 3) = 3,6

Tabel 2.1 Nilai Total Gap Aspek Perilaku

No	ID_Karyawan	NCF	NSF	Np
1	SALES-001	4,5	4,5	4,5
2	SALES-002	4,75	4,5	4,65
3	SALES-003	4	3	3,6

Pada SALES-002 *NFC (Nilaicore faktor),* NSF *(Nilai Secondary Faktor) dan Np(Nilai Perilaku)* mendapat skor tertinggi.

II.3.1. Perhitungan Penentuan Hasil Akhir/Ranking

Hasil akhir dari proses ini adalah ranking dari kandidat yang diajukan untuk mengisi suatu jabatan tertentu. Penentuan ranking mengacu pada hasil perhitungan tertentu. Perhitungan tersebut dapat ditunjukkan pada rumus di bawah ini:

$$Ha = (x)\%Ni + (x)\%Ns + (x)\%Np \quad \text{.....} \quad (5)$$

Di mana :

Ha : Hasil Akhir

Ni : Nilai Kapasitas Intelektual

Ns : Nilai Sikap Kerja

Np : Nilai Perilaku

(x)% : Nilai Persen yang di*input*kan

Sebagai contoh dari rumus untuk perhitungan hasil akhir di atas maka hasil akhir dari karyawan dengan asub aspek PH001 dengan nilai persen = 20%, 30% dan 50%. Dapat dilihat pada proses ini :

Hasil akhir SALES-0001

= (20% x 3,94) + (30% x 4,13) + (50% x 4,5)

= 0,78 + 1,24 + 2,25

= 4,278

Hasil akhir SALES-0002

= (20% x 4,2) + (30% x 3,56) + (50% x 4,65)

= 0,84 + 1,06 + 2,32

= 4,235

Hasil akhir SALES-0003

= (20% x 4,16) + (30% x 3,73) + (50% x 3,6)

= 0,83 + 1,11+ 1,8

= 3,752

II.4. Pengertian *Unified Modeling Language* (UML)

Saat ini piranti lunak luas dan besar lingkupnya, sehingga tidak bisa dibuat asal-asalan. Kesuksesan suatu pemodelan piranti lunak ditentukan oleh tiga unsur, ketiga unsur itu adalah pemodelan (*notation),* proses (*process)* dan tool digunakan. Memahami notasi pemodelan tanpa mengetahui cara pemakaian yang sebenarnya akan membuat proyek gagal. Dan pemahaman terhadap pemodelan dan proses disempurnakan dengan penggunaan tool yang tepat.

Menurut Yuni Sugiarti (2013, h. 34) *Unified Modeling Language* adalah sebuah "bahasa" yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem.

Dengan menggunakan UML kita dapat membuat 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. Tetapi karena UML juga menggunakan *class* dan *operation* dalam konsep dasarnya, maka ia lebih cocok untuk penulisan piranti lunak dalam bahasabahasa berorientasi objek seperti C++, Java, C# atau VB.NET. Walaupun demikian, UML tetap dapat digunakan untuk modeling aplikasi prosedural dalam VB atau C. Seperti bahasa-bahasa lainnya, UML mendefinisikan notasi dan *syntax*/semantik. Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram piranti lunak. Setiap bentuk memiliki makna tertentu, dan UML *syntax* mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan. Notasi UML terutama diturunkan dari 3 notasi yang telah ada sebelumnya: Grady

Booch OOD (Object-Oriented Design), Jim Rumbaugh OMT (*Object Modeling Technique*), dan Ivar Jacobson OOSE (*Object-Oriented Software Engineering*).

Sejarah UML sendiri cukup panjang. Sampai era tahun 1990 seperti kita ketahui puluhan metodologi pemodelan berorientasi objek telah bermunculan di dunia. Diantaranya adalah: metodologi booch [1],metodologi coad [2], metodologi OOSE [3], metodologi OMT [4], metodologi shlaer-mellor [5],metodologi wirfs-brock [6], dsb. Masa itu terkenal dengan masa perang metodologi (*method war*) dalam pendesainan berorientasi objek. Masing-masing metodologi membawa notasi sendiri-sendiri,yang mengakibatkan timbul masalah baru apabila kita bekerjasama dengan group/perusahaan lainyang menggunakan metodologi yang berlainan.

II.4.1. Konsepsi Dasar UML

Dari berbagai penjelasan rumit yang terdapat di dokumen dan buku-buku UML. Abstraksi konsep dasar UML yang terdiri dari *structural classification*, *dynamic behavior*, dan *model management*, bisa kita pahami dengan mudah apabila kita melihat gambar diatas dari *Diagrams*. *Main concepts* bisa kita pandang sebagai iterm yang akan muncul pada saat kita membuatdiagram. Dan view adalah kategori dari diagaram tersebut. (ikc.dinus.ac.id)

Lalu darimana kita mulai untuk menguasai UML, sebenarnya cukup dua hal yang harus kitaperhatikan:

1. Menguasai pembuatan diagram UML.
2. Menguasai langkah-langkah dalam analisa dan pengembangan dengan UML.

Tulisan ini pada intinya akan mengupas kedua hal tersebut.

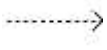
Seperti juga tercantum pada gambar diatas UML mendefinisikan diagram-diagram sebagai berikut:

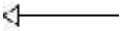
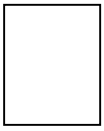
1. Use Case Diagram

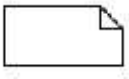
Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, meng-*create* sebuah daftar belanja, dan sebagainya.

Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu. *Use case diagram* dapat sangat membantu bila kita sedang menyusun *requirement* sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang *test case* untuk semua *feature* yang ada pada sistem.

Tabel 2.2 *Diagram Use Case*

NO	GAMBAR	NAMA	KETERANGAN
1		*Actor*	Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.
2		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*).

3		*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
4		*Include*	Menspesifikasikan bahwa *use case* sumber secara *eksplisit*.
5		*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
6		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor
9		*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).

10		*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi

(Sumber:Yuni Sugiarti 2013,34)

2. *Activity Diagram*

Activity diagram menyediakan analis dengan kemampuan untuk memodelkan proses dalam suatu sistem informasi. *Activity diagram* dapat digunakan untuk alur kerja model, *use case* individual, atau logika keputusan yang terkandung dalam metode individual3. *Activity diagram* juga menyediakan pendekatan untuk proses pemodelan paralel. *Activity diagram* lebih lanjut .

Pada dasarnya, diagram aktifitas canggih dan merupakan diagram aliran data yang terbaru. Secara teknis, diagram aktivitas menggabungkan ide-ide proses pemodelan dengan teknik yang berbeda termasuk model acara, *statecharts*, dan Petri Nets.

3. *Package Diagram*

Package diagram utamanya digunakan untuk mengelompokkan elemen diagram UML yang berlainan secara bersama-sama ke dalam tingkat pembangunan yang lebih tinggi yaitu berupa sebuah paket. Diagram paket pada dasarnya adalah diagram kelas yang hanya menampilkan paket, disamping kelas, dan hubungan ketergantungan, disamping hubungan khas yang ditampilkan pada diagram kelas.

Sebagai contoh, jika kita memiliki sistem pendaftaran untuk kantor dokter, mungkin masuk akal untuk kelompok kelas pasien dengan kelas sejarah medis pasien bersama-sama untuk membentuk paket kelas pasien. Selain itu, dapat berguna untuk membuat paket perawatan yang mengandung gejala penyakit, penyakit, dan obat-obatan khas yang diresepkan untuk mereka.

4. Sequence Diagram

Sequence diagram menjelaskan interaksi objek yang disusun berdasarkan urutan waktu. Secara mudahnya *sequence diagram* adalah gambaran tahap demi tahap yang seharusnya dilakukan untuk menghasilkan sesuatu sesuai dengan *use case diagram.*

II.4.2. Tujuan UML *(Unified Modeling Language)*

Adapun tujuan dari penggunaan UML *(Unified Modeling Languarge)*

1. Memberikan bahasa pemodelan yang bebas dari berbagai bahasa pemograman dan proses rekayasa.
2. Menyatukan praktek – praktek terbaik yang dapat terdalam pemodelan.
3. Memberikan model yang siap pakai, bahasa pemodelan visual yang ekspresif untuk mengembangkan dan saling menukar model dengan mudah dan dimengerti secara umum.
4. *UML* bisa juga berfungsi sebagai sebuah *(blue print)* cetak biru karena sangat lengkap dan *detail.* Dengan cetak biru ini maka akan bisa diketahui informasi

secara *detail* dan *coding* program atau bahkan membaca program dan menginterpretasikan kembali kedalam bentuk diagram (*reserve engineering*).

II.5. Pengertian Basis Data (*Database*)

Basis data (database) merupakan suatu data yang terintegrasi, diorganisasikan, dan disimpan dalam suatu cara yang baik tergantung pada aspek desain dan aspek realnya.

Menurut Marlinda (2005, h. 1) menyatakan bahwa “basis data adalah suatu susunan kumpulan data oprasional lengkap dari suatu organisasi/perusahaan yang terorganisir/dikelola dan disimpan secara terintegrasi dengan menggunakan metode tertentu menggunakan komputer sehingga mampu menggunakan metode tertentu menggunakan komputer sehingga mampu menyediakan informasi optimal yang diperlukan pemakainya”.

Menurut Sutabri (2005, h. 161) menyatakan bahwa “database merupakan suatu kumpulan data terhubung (*interrelated data*) yang disimpan secara bersama-sama pada suatu media, tanpa mengatap satu sama lain tidak perlu suatu kerangkapan data (*controlled redundancy)* dengan cara tertentu sehingga mudah digunakan atau ditampilkan kembali”.

II.5.1 Model Data

Model data merupakan sejumlah konsep yang digunakan untuk membuat deskripsi struktur basis data, dengan deskripsi struktur basis data dapat ditemukan jenis data, hubungan dan konstrain data yang harus ditangani. Kebanyakan model

data juga membuat spesifikasi untuk operasi dasar dalam pengaksesan dan pembaharuan data pada basis. Model data dapat duikelompokkan berdasarkan konsep pembuatan deskripsi struktur basis data, yaitu model data konseptual dan model data fisikal.

1. Model data konseptual

   Menyajikan tentang bagaimana pemakai basis data memandang atau memperlakukan data. Dalam model data konseptual digunakan konsep entity, atribut dan hubungan.

2. Model data fisikal

   Merupakan konsep bagaimana deskripsi detail data disimpan dalam komputer dengan menyajikan informasi tentang format rekaman, urutan rekaman dan jalur pengaksesan data. Informasi jalur pengaksesan dan merupakan struktur yang dapat membuat pencarian rekaman data lebih efisien.

II.5.2. *Hierarki* Data Dalam *Database*

Database merupakan salah satu komponen yang penting dalam sistem informasi, karena merupakan basis (dasar) dalam menyediakan informasi bagi para pemakai.

1. Bit

Bit merupakan bagian data terkecil, dapat berupa karakter numerik, huruf maupun karakter-karakter khusus yang membentuk suatu item data, dimana kumpulan karakter membentuk satu *field.*

2. *Byte*

Byte merupakan sistem binner yang terdiri atas dua macam nilai, yaitu 0 dan 1. Sistem binner merupakan dasar yang dapat digunakan untuk komunikasi antara manusia dan mesin.

3. *Field*

Suatu *field* menggambarkan suatu atribut *record* yang menunjukan suatu item dari data seperti nama, alamat, dimana kumpulan *field* membentuk suatu *record.*

4. *Record*

Suatu *record* menggambarkan satu kesatuan data yang sejenis, dimana kumpulan dari *file-file* membentuk *database.*

5. *File*

Suatu *file* menggambarkan kumpulan dari beberapa *record* yang dapat menampung data-data.

6. *Database*

Suatu *database* menggambar data yang saling berhubungan antara satu dengan yang lainnya.

II.6. Pengertian Php

PHP adalah bahasa pemrograman script yang paling banyak dipakai saat ini. PHP banyak dipakai untuk memrogram situs web dinamis, walaupun tidak tertutup kemungkinan digunakan untuk pemakaian lain. Contoh terkenal dari aplikasi PHP adalah forum (phpBB) dan MediaWiki (*software* di belakang Wikipedia). PHP juga dapat dilihat sebagai pilihan lain dari

ASP.NET/C#/*VB.NET Microsoft, ColdFusion Macromedia, JSP/Java Sun Microsystems, dan CGI/Perl*. Contoh aplikasi lain yang lebih kompleks berupa CMS yang dibangun menggunakan PHP adalah Mambo, Joomla, Postnuke, Xaraya, dan lain-lain (Adis Lena Kusuma Ratna;2014)

II.6.1. Kelebihan PHP dari bahasa pemrograman lain

Bahasa pemrograman PHP adalah sebuah bahasa script yang tidak melakukan sebuah kompilasi dalam penggunaanya. *Web Server* yang mendukung PHP dapat ditemukan dimana mana dari mulai apache, IIS, Lighttpd, hingga Xitami dengan konfigurasi yang relatif mudah.Dalam sisi pengembangan lebih mudah, karena banyaknya milis milis dan developer yang siap membantu dalam pengembangan. Dalam sisi pemahamanan, PHP adalah bahasa scripting yang paling mudah karena memiliki referensi yang banyak. PHP adalah bahasa open source yang dapat digunakan di berbagai mesin (Linux, Unix, Macintosh, Windows) dan dapat dijalankan secara runtime melalui console serta juga dapat menjalankan perintah perintah system (AdiL ena Kusuma Ratna;2014)

II.7. Pengertian *MySQL*

MySQL adalah sebuah perangkat lunak sistem manajemen basis data *SQL* (bahasa Inggris: *database management system*) atau *DBMS* yang *multithread, multiuser*, dengan sekitar 6 juta instalasi di seluruh dunia. *MySQL* AB membuat *MySQL* tersedia sebagai perangkat lunak gratis dibawah lisensi GNU *General Public License (GPL)*, tetapi mereka juga menjual dibawah

lisensi komersial untuk kasuskasus dimana penggunaannya tidak cocok dengan penggunaan GPL (Adis Lena Kusuma Ratna;2014)

II.7.1. *Relational Database Management System (RDBMS)*

MySQL adalah *Relational Database Management System (RDBMS)* yang didistribusikan secara gratis dibawah lisensi GPL *(General Public License).* Dimana setiap orang bebas untuk menggunakan *MySQL*, namun tidak boleh dijadikan produk turunan yang bersifat komersial. *MySQL* sebenarnya merupakan turunan salah satu konsep utama dalam database sejak lama, yaitu *SQL (Structured Query Language). SQL* adalah sebuah konsep pengoperasian database, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis. Keandalan suatu sistem *database (DBMS)* dapat diketahui dari cara kerja *optimizer* nya dalam melakukan proses perintah perintah *SQL*, yang dibuat oleh user maupun programprogram aplikasinya. Sebagai database server, MySQL dapat dikatakan lebih unggul dibandingkan database server lainnya dalam query data. Hal ini terbukti untuk *query* yang dilakukan oleh *single user*, kecepatan *query MySQL* bisa sepuluh kali lebih cepat dari *PostgreSQL* dan lima kali lebih cepat dibandingkan *Interbase* (Adis Lena Kusuma Ratna;2014)