BAB 2

LANDASAN TEORI

2.1 Penilaian Kinerja

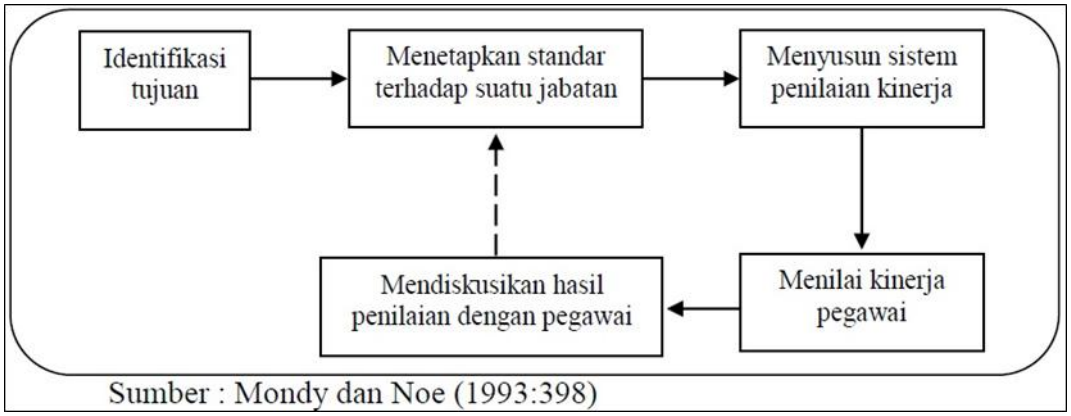
Menurut Siagian (1995:225-226) penilaian prestasi kerja adalah suatu pendekatan dalam melakukan prestasi kerja para pegawai yang didalamnya terdapat berbagai faktor seperti :

1. Penilaian dilakukan pada manusia sehingga disamping memiliki kemampuan tertentu juga tidak luput dari berbagai kelemahan dan kekurangan;
2. Penilaian yang dilakukan pada serangkaian tolak ukur tertentu yang realistik, berkaitan langsung dengan tugas seseorang serta kriteria yang ditetapkan dan diterapkan secara obyektif.

Hasil penilaian harus disampaikan kepada pegawai yang dinilai dengan tiga maksud:

1. Apabila penilaian tersebut positif maka penilaian tersebut menjadi dorongan kuat bagi pegawai yang bersangkutan untuk lebih berprestasi lagi pada masa yang akan datang sehingga kesempatan meniti karier lebih terbuka baginya.
2. Apabila penilaian tersebut negatif maka pegawai yang bersangkutan mengetahui kelemahannya dan dengan sedemikian rupa mengambil berbagai langkah yang diperlukan untuk mengatasi kelemahan tersebut.
3. Hasil penilaian yang dilakukan secara berkala itu terdokumentasi dengan baik dalam arsip kepegawaian setiap pegawai sehingga tidak ada informasi yang hilang, baik sifatnya menguntungkan maupun merugikan pegawai bersangkutan.

Proses penyusunan penilaian kinerja menurut Mondy dan Noe (1993:398) terbagi dalam beberapa tahapan kegiatan yang ditunjukkan dalam gambar di bawah ini :



Gambar 0.1 Tahapan Penilaian Kinerja

Langkah pertama yang harus dilakukan dalam menyusun sistem penilaian kinerja yaitu harus digali terlebih dahulu tujuan yang ingin dicapai oleh organisasi dengan adanya sistem penilaian kinerja yang akan disusun.

Langkah yang kedua, menetapkan standar yang diharapkan dari suatu jabatan, sehingga akan diketahui dimensi-dimensi apa saja yang akan diukur dalam penilaian kinerja. Dimensi-dimensi tersebut tentunya harus sangat terkait dalam pelaksanaan tugas pada jabatan itu. Tahap ini biasanya dapat dilakukan dengan menganalisa jabatan atau menganalisa uraian tugas masing-masing jabatan.

Setelah tujuan dan dimensi yang akan diukur dalam penilaian kinerja diketahui, maka langkah selanjutnya yaitu menentukan desain yang sesuai untuk mencapai tujuan yang diharapkan. Penentuan desain penilaian kinerja ini harus selalu dikaitkan dengan tujuan penilaian.

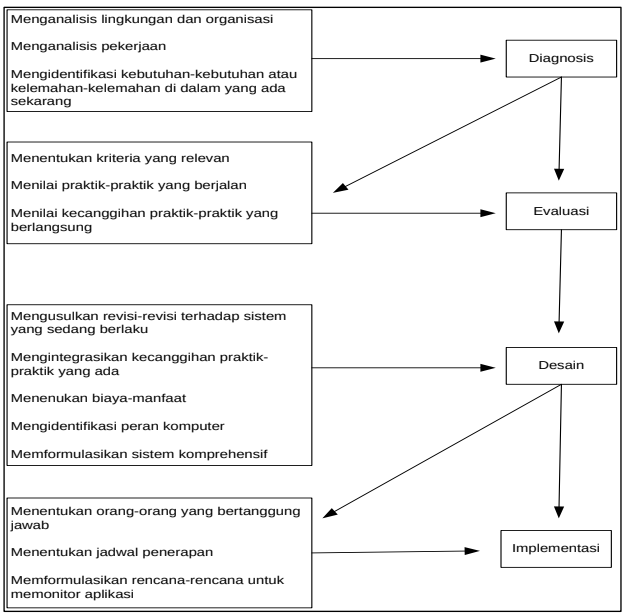
Langkah berikutnya adalah melakukan penilaian kinerja terhadap pegawai yang menduduki suatu jabatan. Hasil dari penilaian kinerja, selanjutnya dianalisa

dan dikomunikasikan kembali kepada pegawai yang dinilai agar mereka mengetahui kinerjanya selama ini serta mengetahui kinerja yang diharapkan oleh organisasi. Evaluasi terhadap sistem penilaian kinerja yang telah dilakukan juga dilaksanakan pada tahap ini. Apabila penilaian kinerja tersebut sudah dapat mencapai tujuan dari diadakannya penilaian kinerja atau belum. Apabila ternyata belum, maka harus dilakukan revisi atau mendesain ulang sistem penilaian kinerja.

Empat langkah umum untuk meningkatkan efektivitas manajemen sumber daya manusia:

1. Diagnosis permasalahan.

   Manajer sumber daya manusia dan manajemen puncak harus mendiagnosis permasalahan dalam mengelola sumber daya manusia mereka. Diagnosis membutuhkan adanya gambaran mengenai organisasi, pekerjaan-pekerjaan yang ada didalamnya, karyawan, lingkungan, dan faktor-faktor lainnya yang mempengaruhi pendayagunaan sumberdaya manusia di dalam organisasi.



Gambar 0.2 Pendekatan praktis terhadap Manajemen Sumber Daya Manusia
(Sumber : Simamora (2001 :50))

2. Evaluasi praktik yang ada

   Evaluasi praktik yang ada. Manajer dan pakar sumber daya manusia haruslah mengevaluasi praktik personalia dari bagian organisasi yang lain yang saat ini berjalan. Mereka haruslah menilai manfaat praktik dan prosedur ini terhadap karyawan, biaya, dan efektivitasnya dalam mencapai tujuan organisasi. Penilaian atas praktik-praktik yang ada haruslah mempertimbangkan kriteria-kriteria yang berlaku.

3. Desain sistem manajemen sumber daya manusia

   Desain sistem manajemen sumber daya manusia. Manajer dan pakar sumber daya manusia haruslah mendesain sistem manajemen sumber daya manusia yag saling berkaitan. Mereka harus mengoreksi kelemahan di dalam sistem tersebut.

4. Implementasi sistem

   Implementasi sistem. Desain sistem tidak akan dapat dilaksanakan dalam kevakuman. Langkah terakhir adalah mempertimbangkan siapa yang akan menerapkan kebijakan bagaimana sumber daya akan didistribusikan, bagaimana implementasi akan dilaksanakan, dan bagaimana memantau pengelolaan sumber daya manusia.

2.2 Preference Ranking Organization Method For Enrichment Evaluation

Preference Ranking Organization Method For Enrichment Evaluation (Promethee) merupakan salah satu ranking dalam *Multiple Criteria Decision Making* (MCDM). Pengertian dari metode Promethee adalah suatu metode menentuan urutan (prioritas) dalam analisa multikriteria. Masalah pokoknya adalah kesederhanaan, kejelasan, dan kestabilan. Dugaan dari dominasi kriteria

yang digunakan dalam promethee adalah penggunaan nilai dalam hubungan *outranking*. Semua parameter yang dinyatakan mempunyai pengaruh nyata menurut pandangan ekonomi (Suryadi, 2002).”

2.2.1 Karakteristik Promethee

Prinsip yang digunakan adalah penetapan prioritas alternatif yang telah ditetapkan berdasarkan pertimbangan ($\forall i \mid fi(,) \rightarrow \Re$ [real world]), dengan kaidah dasar:

Max{$f_1(x)$, $f_2(x)$, $f_3(x)$)…, $f_j(x)$ …, $f_k(x) \mid x \in \Re$} .....(0.1)

Dimana K adalah sejumlah kumpulan alternatif dan fi (i = 1,2,3,…, k) merupakan nilai atau ukuran relatif kriteria untuk masing-masing alternatif. Dalam aplikasinya sejumlah kriteria telah ditetapkan untuk menjelaskan K yang merupakan penilaian dari $\Re$ (*real world*). Promethee termasuk dalam keluarga dari metode *outranking* yang dikembangkan oleh B.Roy dan meliputi 2 fase:

1. Membangun hubungan *outranking* dari K.
2. Eksploitasi dari hubungan ini memberikan jawaban optimasi kriteria dalam paradigm permasalahan multikriteria.

Pada tahap pertama, nilai hubungan *outranking* berdasarkan pertimbangan dominasi masing-masing kriteria. Indeks preferensi ditentukan dan nilai *outranking* secara grafis disajikan berdasarkan preferensi dari pembuat keputusan.

Pada tahap kedua, eksploitasi dilakukan dengan mempertimbangkan nilai *Leaving Flow* dan *Entering Flow* pada grafik nilai *outrangking*. Urutan parsial untuk Promethee I dan urutan lengkap pada Promethee II. Data dasar untuk evaluasi dengan metode Promethee terdapat pada Tabel 0.1

Tabel 0.1 Data Dasar Analisa Promethee.

	$f_1(\ ,\ )$	$f_2(\ ,\ )$	…	$f_i(\ ,\ )$	…	$f_k(\ ,\ )$
a_1	$f_1(a_1)$	$f_2(a_1)$	…	$f_i(a_1)$	…	$f_k(a_1)$
a_2	$f_1(a_2)$	$f_2(a_2)$	…	$f_i(a_2)$	…	$f_k(a_2)$
…	…	…	…	…	…	…
a_i	$f_1(a_i)$	$f_2(a_i)$	…	$f_i(a_i)$	…	$f_k(a_i)$
…	…	…	…	…	…	…
a_n	$f_1(a_n)$	$f_2(a_n)$	…	$f_i(a_n)$	…	$f_k(a_n)$

Keterangan :

1. a_1, a_2, a_i, a_n : a alternatif potensial.
2. f_1, f_2, f_i, f_k : f kriteria evaluasi.

2.2.2 Dominasi Kriteria

Penyampaian intensitas (*P*) dari preferensi alternatif a terhadap alternatif b sedemikian rupa sehingga:

a. P(a,b) = 0, berarti tidak ada beda (*indifferent*) antara a dan b, atau tidak ada preferensi dari a lebih baik dari b.

b. P(a,b) ~ 0, berarti lemah preferensi dari a lebih baik dari b.

c. P(a,b) ~ 1, berarti kuat preferensi dari a lebih baik dari b kuat.

d. P(a,b) = 1, berarti mutlak preferensi dari a lebih baik dari b.

Dalam metode ini, fungsi preferensi sering kali menghasilkan nilai fungsi yang berbeda antara dua evaluasi sehingga:

P(a,b) = P(f(a) – f(b)).....(0.2)

Untuk semua kriteria, suatu alternatif akan dipertimbangkan memiliki nilai kriteria yang lebih baik ditentukan oleh niali f dan nilai akumulasi dari nilai ini menentukan nilai preferensi atas masing-masing alternatif yang akan dipilih.

2.2.3 Rekomendasi fungsi preferensi untuk keperluan sistem

Menurut Kadarsah (1998:148), dalam metode Promethee terdapat enam fungsi preferensi kriteria, yaitu:

1. Kriteria biasa (*Usual Criterian*)
2. Kriteria quasia (*Quasi Criterian*)
3. Kriteria *linear*
4. Kriteria *level*
5. Kriteria dengan preferensi *linear* dan area yang tidak berbeda
6. Kriteria *gaussian*

Hal ini tentu saja tidak mutlak, tetapi bentuk ini cukup baik untuk beberapa kasus. Untuk memberikan gambaran yang lebih baik terdapat area yang tidak sama, digunakan fungsi selisih nilai kriteria antara alternatif H(d) dimana hal ini mempunyai hubungan langsung pada fungsi preferensi. Penjelasan masing-masing kriteria tersebut adalah sebagai berikut:

1. Kriteria biasa (*Usual Criterian*)

$$\boldsymbol{H(d) = \begin{cases} 0 \; Jika \; d = 0 \\ 1 \; jika \; d \neq 0 \end{cases}} \quad \textbf{.....(0.3)}$$

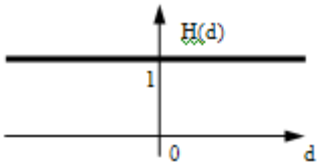
dimana

d : selisih nilai kriteria { d = f(a) - f(b) }

H(d) : fungsi selisih kriteria antar alternatif.

Pada kasus ini tidak ada beda (sama penting) antara a dan b jika f(a) = f(b); apabila nilai kriteria pada masing-masing alternatif memiliki nilai berbeda, pembuatan keputusan membuat preferensi mutlak untuk alternatif yang memiliki nilai lebih baik. Untuk memilih kasus preferensi pada kriteria biasa, ilustrasinya dapat dilihat dari perlombaan renang, seorang peserta dengan peserta lainnya akan

memiliki peringkat yang mutlak berbeda walaupun hanya dengan selisih nilai (waktu) teramat kecil, dan dia akan memiliki peringkat yang sama jika dan hanya jika waktu tempuhnya sama atau selisih nilai diantara keduanya sebesar nol. Fungsi H(d) untuk fungsi preferensi ini ditunjukkan pada Gambar 0.3



Gambar 0.3 Kriteria biasa.

2. Kriteria quasi (*Quasi Criterian*)

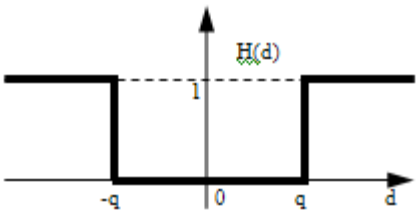
$$\boldsymbol{H(d)} = \begin{cases} \boldsymbol{0\ Jika - q = d = q} \\ \boldsymbol{1\ jika\ d} < -q\ atau\ d > q \end{cases} \quad \text{.....(0.4)}$$

dimana:

d : selisih nilai kriteria { d = f(a) - f(b) }

H(d) : fungsi selisih nilai kriteria antara alternatif.

Parameter (q) : harus merupakan nilai yang tetap, q = 0.



Gambar 0.4 Kriteria quasi.

Kriteria ini memiliki alternatif preferensi yang sama penting selama selisih atau nilai H(d) dari masing-masing alternatif untuk kriteria tertentu tidak melebihi nilai q, dan apabila selisih hasil evaluasi untuk masing-masing alternatif melebihi nilai q maka akan terjadi bentuk preferensi mutlak. Fungsi H(d) untuk fungsi preferensi ini ditunjukkan pada Gambar 0.4

Misalnya seorang akan dipandang mutlak lebih kaya apabila selisih nilai kekayaannya lebih besar dari Rp10.000.000 dan apabila selisih kekayaannya kurang dari Rp 10.000.000 dipandang sama kaya.

3. Kriteria *linear*

$$\boldsymbol{H(d)} = \begin{cases} \boldsymbol{d/p\ \ Jika - p = d = p} \\ \boldsymbol{1\ \ Jika\ d} < -p\ atau\ d > p \end{cases} \quad \text{.....(0.5)}$$

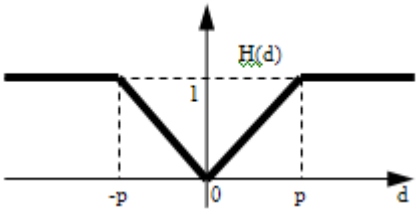
dimana :

d : selisih nilai kriteria { d = f(a) - f(b) }

p : nilai kecenderungan atas.

H(d) : fungsi selisih nilai kriteria antar alernatif.

Kriteria ini menjelaskan bahwa selama nilai selisih memiliki nilai yang lebih rendah dari p, preferensi dari pembuat keputusan meningkat secara *linear* dengan nilai d. Jika nilai d lebih besar dibandingkan dengan nilai p, maka terjadi preferensi mutlak. Fungsi H(d) untuk fungsi preferensi ini ditunjukkan pada Gambar 0.5



Gambar 0.5 Kriteria *linear*.

Misal akan terjadi preferensi dalam hubungan *linear* kriteria kecerdasan seseorang dengan cara lain apabila nilai seseorang berselisih dibawah 30, apabila diatas 30 poin maka mutlak dikatakan orang itu lebih cerdas dibandingkan dengan orang lain.

4. Kriteria *level*

$$H(d) = \begin{cases} 0 \ jika \ |d| = q \\ 0,5 \ jika \ q < |d| = p \\ 1 \ jika \ p < |d| \end{cases} \quad \text{.....(0.6)}$$

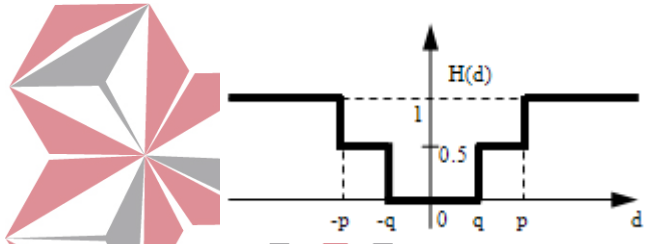
dimana:

p : nilai kecenderungan atas.

H(d) : fungsi selisih kriteria antar alternatif.

Parameter (q) : harus merupakan nilai yang tetap, q = 0

Dalam kasus ini kecenderungan tidak berbeda q dan kecenderungan preferensi p ditentukan secara simultan. Jika d berada di antara nilai q dan p, hal ini berarti situasi preferensi yang lemah (H(d) = 0.5).



Gambar 0.6 Kriteria *level*.

Gambar 0.6 menjelaskan pembuat keputusan telah menentukan kedua kecenderungan untuk kriteria ini. Bentuk kriteria level ini dapat dijelaskan misalnya dalam penetapan nilai preferensi jarak tempuh anata kota. Misalnya jarak antara Surabaya-Bromo sebesar 60 km, Bromo-Kalibaru sebesar 68 km. Kalibaru-Ijen sebesar 45 km, Bromo-Ijen 133 km. Dan telah ditetapkan bahwa selisih dibawah 10 km maka dianggap jarak antar kota tersebut adalah tidak berbeda, selisih jarak sebesar 10-30 km relative berbeda dengan preferensi yang lemah, sedangkan selisih diatas 30 km diidentifikasi memiliki preferensi mutlak berbeda.

Dalam kasus ini, selisih jarak antara Surabaya-Bromo dan Bromo-Kalibaru

dianggap tidak berbeda (H(d) = 0) karena selisih jaraknya dibawah 10 km, yaout (68-60) km = 8 km, sedangkan preferensi jarak antara Bromo_kalibaru dan Kalibaru-Ijen dianggap berbeda dengan preferensi lemah (H(d)=0.5) karena memiliki selisih yang berada pada interval 10-30 km, yaitu (68-45) km = 23 km. Dan terjadi preferensi mutlak (H(d) = 1) antara jarak Bromo_Ijen dan Kalibaru-Ijen karena memiliki selisih jarak lebih dari 30 km.

5. Kriteria dengan preferensi *linear* dan area yang tidak berbeda

$$\boldsymbol{H(d)} = \begin{cases} \boldsymbol{0 \ jika \ |d| = q} \\ \boldsymbol{(|d| - q)/(p - q) \ jika \ q < |d| = p} \\ \boldsymbol{1 \ jika \ p < |d|} \end{cases} \quad \text{.....(0.7)}$$

dimana:

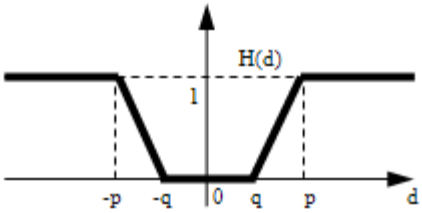
d : selisih nilai kriteria { d = f(a) - f(b) }

H(d) : fungsi selisih nilai kriteria antar alternatif.

Parameter (p) : nilai kecendrungan atas.

Parameter (q) : harus merupakan nilai yang tetap, q = 0

Pada kasus ini, pengambil keputusan mempertimbangkan peningkatan preferensi secara *linear* dan tidak berbeda hingga preferensi mutlak dalam area antara dua kecenderungan q dan p. Dua parameter tersebut telah ditentukan dimana fungsi H adalah hasil perbandingan antar alternatif, seperti yang ditunjukkan pada Gambar 0.7

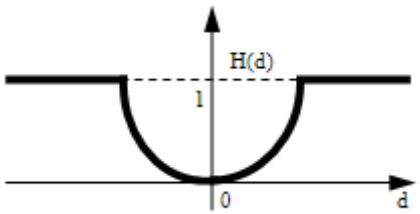


Gambar 0.7 Kriteria preferensi *linear* dan area yang tidak berbeda.

6. Kriteria *gaussian*

$$H(d) = 1 - \exp(-d^2/2\sigma^2) \quad \text{.....(0.8)}$$

Fungsi ini bersyarat apabila telah ditentukan nilai σ, dimana dapat dibuat berdasarkan distribusi normal dalam statistic. Fungsi ini ditunjukkan seperti pada Gambar 0.8



Gambar 0.8 Kriteria *gaussian*.

2.2.4 Penentuan Tipe Preferensi

Seperti telah disebutkan diatas, maka proses penentuan preferensi merupakan langkah yang penting sehingga saat perhitungan indeks preferensi dapat representatif terhadap permasalahan. Dalam membantu penentuan tingkat preferensi dapat ditunjukkan pada Tabel 0.2.

Tabel 0.2 Penentuan Tingkat Preferensi.

Pertimbangan	Tingkat Fungsi Preferensi					
	I	II	III	IV	V	VI
Akurasi	Kasar	Kasar	Akurat	Kasar	Akurat	Akurat
Kecenderungan tidak berbeda |d| < q	Tidak	Ya	Tidak	Ya	Ya	Tidak
Kecenderungan kokoh mutlak |d| < q	Tidak	Tidak	Tidak	Ya	Ya	Tidak
Distribusi normal	Mungkin	Mungkin	Mungkin	Mungkin	Mungkin	Ya

2.2.5 Indeks Preferensi Multikriteria

Tujuan pembuatan keputusan adalah menetapkan fungsi preferensi Pi dan πi untuk semua kriteria fi (i = 1,2,3,….,k) dari masalah optimasi kriteria majemuk.

Bobot (weight) xi merupakan ukuran relatif dari kepentingan kriteria fi, jika semua kriteria memiliki nilai kepentingan yang sama dalam pengambilan maka semua nilai bobot adalah sama.

Indeks preferensi multikriteria ditentukan berdasarkan rata-rata bobot dari fungsi preferensi Pi.

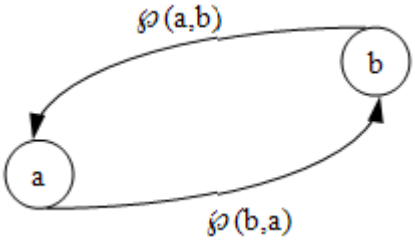
$$\wp(\boldsymbol{a},\boldsymbol{b}) = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \boldsymbol{\pi}(\boldsymbol{a},\boldsymbol{b}) : \forall\, \boldsymbol{a},\boldsymbol{b} \in \boldsymbol{A} \quad \text{.....(0.9)}$$

dimana:

1. $\wp$(a,b), merupakan intensitas preferensi pembuat keputusan yang menyatakan bahwa alternatif a lebih baik dari alternatif b dengan pertimbangan secara simultan dari seluruh kriteria.
2. $\wp$(a,b) = 0, menunjukkan preferensi yang lemah untuk alternatif a labih dari alternatif b berdasarkan semua kriteria.
3. $\wp$(a,b) = 1, menunjukkan preferensi yang kuat untuk alternatif a lebih dari altenatif b berdasarkan semua kriteria.

Index preferensi ditentukan berdasarkan nilai hubungan *outranking* pada sejumlah kriteria dari masing-masing alternatif.

Hubungan ini dapat ditunjukkan sebagai grafik nilai *outranking*, node-nodenya merupakan alternatif berdasarkan penilaian kriteria tertentu. Diantaranya dua node (alternatif) a dan b, merupakan garis lengkung yang mempunyai nilai $\wp$(a,b) dan $\wp$(b,a) (tidak ada hubungan khusus antara $\wp$(a,b) dan $\wp$(b,a)). Hal ini ditunjukkan pada Gambar 0.9 di halaman 14.



Gambar 0.9 Hubungan antar Node

2.2.6 Arah Preferensi Multikriteria

Arah preferensi terbagi menjadi dua arah yaitu *Leaving Flow* (LF) dan *Entering Flow* (EF). *Leaving Flow* merupakan ukuran dari karakter *outranking* a, sedangkan *Entering Flow* merupakan ukuran karakter a yang di *outrank*. Secara sistematis dapat ditentukan *Leaving Flow* dengan persamaan sebagai berikut:

$$\boldsymbol{\phi^{+}(a) = \frac{1}{n-1} \sum_{x \epsilon A} \wp (x, b)} \quad \textbf{.....(0.10)}$$

Adapun persamaan *Entering Flow* adalah sebagai berikut:

$$\boldsymbol{\phi^{-}(a) = \frac{1}{n-1} \sum_{x \epsilon A} \wp (x, b)} \quad \textbf{.....(0.11)}$$

Adapun persamaan *Net Flow* adalah sebagai berikut:

$$\boldsymbol{\phi(a) = \phi^{+}(a) - \phi^{-}(a)} \quad \textbf{.....(0.12)}$$

2.2.7 Perangkingan dalam Promethee

Dalam metode Promethee proses perankingan dilakukan melalui dua perankingan yaitu Promethee I (Promethee parsial) dan Promethee II (*Promethee complete*). Perankingan Promethee I didasarkan pada masing-masing nilai *Leaving Flow* dan *Entering Flow*. Semakin besar nilai *Leaving Flow* dan semain kecil nilai *Entering Flow* maka alternatif semakin baik. Jika nilai ranking *Leaving Floe* dan *Entering Flow* sama maka hasil rangking Promethee I menjadi solusi metode Promethee. Tetapi, jika sebaliknya maka proses harus dilanjutkan ke

Promethee II. Promethee II didasarkan pada nilai dari *Netflow*. Semakin besar nilai *Net Flow* maka semakin tinggi rangkingnya.

2.3 Konsep Sistem Informasi

Sistem informasi (Kendall: 2003) didefinisikan oleh Robert A. Leitch dan K. Roscoe Davis sebagai berikut: "Sistem informasi adalah suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat manajerial dan kegiatan strategi dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan - laporan yang diperlukan."

2.3.1 Blok Masukan

Masukan atau Input mewakili data yang masuk ke dalam sistem informasi. Masukan disini termasuk metode-metode dan media untuk menangkap data yang akan dimasukkan, yang dapat berupa dokumen-dokumen dasar.

2.3.2 Blok Masukan

Blok ini terdiri dari kombinasi prosedur, logika dan model matematik yang akan memanipulasi data input dan data yang tersimpan di basis data dengan cara yang sudah ditentukan untuk menghasilkan keluaran yang diinginkan.

2.3.3 Blok Keluaran

Produk dari sistem informasi adalah keluaran yang merupakan inforrmasi yang berkualitas dan dokumentasi yang berguna untuk semua tingkatan manajemen serta semua pemakai sistem.

2.3.4 Blok Teknologi

Teknologi merupakan "kotak alat" (*toolbox*) dalam sistem informasi. Teknologi digunakan untuk menerima input, menjalankan model, menyimpan dan

mengakses data, menghasilkan dan mengirimkan keluaran dan membantu pengendalian dari sistem secara keseluruhan.

2.3.5 Basis Data

Basis data (*database*) merupakan kumpulan dari data yang saling berhubungan satu dengan lainnya, tersimpan di perangkat keras komputer dan digunakan perangkat lunak untuk memanipulasinya. Data perlu disimpan di dalam basis data untuk keperluan penyediaan informasi lebih lanjut. Data di dalam basis data perlu diorganisasikan sedemikian rupa, supaya informasi yang dihasilkan berkualitas. Organisasi basis data yang baik juga berguna untuk efisiensi kapasitas penyimpannya. Basis data diakses atau dimanipulasi dengan menggunakan perangkat lunak paket yang disebut dengan DBMS (*Database Management Systems*).

2.3.6 Blok Kendali

Banyak hal yang dapat merusak sistem informasi, seperti misalnya bencana alam, api, temperatur, air, debu, kecurangan-kecurangan, kegagalan-kegagalan sistem itu sendiri, kesalahan-kesalahan, ketidak-efisienan, sabotase, dan lain sebagainya. Beberapa pengendalian perlu dirancang dan diterapkan untuk meyakinkan bahwa hal-hal yang dapat merusak sistem dapat dicegah ataupun bila terlanjur terjadi kesalahan-kesalahan dapat langsung diatasi.

2.4 Data Flow Diagram

DFD sering digunakan untuk menggambarkan suatu sistem yang telah ada atau sistem baru yang akan dikembangkan secara logika tanpa mempertimbangkan lingkungan fisik dimana data tersebut mengalir. DFD

merupakan alat yang digunakan pada metodologi pengembangan sistem yang terstruktur dan dapat mengembangkan arus data di dalam sistem dengan terstruktur dan jelas.

2.4.1 Simbol-simbol yang digunakan dalam DFD

1. External Entity atau Boundary

   External entity atau kesatuan luar merupakan kesatuan di lingkungan luar sistem yang dapat berupa orang, organisasi atau sistem lainnya yang berada di lingkungan luarnya yang akan memberikan input atau menerima output dari sistem. *External entity* disimbolkan dengan notasi kotak.

2. Arus Data

   Arus Data (*data flow*) di DFD diberi simbol panah. Arus data ini mengalir di antara proses, simpanan data (*data store*) dan kesatuan luar (*external entity*). Arus data ini menunjukkan arus data yang dapat berupa masukan untuk sistem atau hasil dari proses sistem.

3. Proses

   Suatu proses adalah kegiatan yang dilakukan oleh orang, mesin, atau komputer dari hasil suatu arus data yang masuk ke dalam proses untuk menghasilkan arus data yang akan keluar dari proses. Simbol proses berupa lingkaran atau persegi panjang bersudut tumpul.

4. Simpanan Data

   Simpanan data di DFD disimbolkan dengan sepasang garis horizontal paralel yang tertutup di salah satu ujungnya. Simpanan data merupakan simpanan dari data yang dapat berupa hal-hal sebagai berikut, sebagai gambaran:

   1 Suatu file atau *database* di sistem komputer.

2 Suatu arsip atau catatan manual.

3 Suatu kotak tempat data di meja seseorang.

4 Suatu tabel acuan manual.

5. Context Diagram

Context Diagram merupakan langkah pertama dalam pembuatan DFD. Pada *context diagram* dijelaskan sistem apa yang dibuat dan *eksternal entity* apa saja yang terlibat. Dalam context diagram harus ada arus data yang masuk dan arus data yang keluar.

6. Data Flow Diagram Level 0

DFD level 0 adalah langkah selanjutnya setelah *context diagram*. Pada langkah ini, digambarkan proses-proses yang terjadi dalam sistem informasi.

7. Data Flow Diagram Level 1

DFD Level 1 merupakan penjelasan dari DFD level 0. Pada proses ini dijelaskan proses apa saja yang dilakukan pada setiap proses yang terdapat di DFD level 0.

2.5 Entity Relational Diagram

Entity Relational Diagram (ERD) merupakan penggambaran hubungan antara beberapa entity yang digunakan untuk merancang *database* yang akan diperlukan.

2.6 Sistem Basis Data

Menurut Marlinda (2004: 1) Sistem Basis Data adalah suatu sistem menyusun dan mengelola *record* menggunakan komputer untuk menyimpan atau merekam serta memelihara data operasional lengkap sebuah organisasi atau perusahaan sehingga mampu menyediakan informasi yang optimal yang

diperlukan pemakai untuk proses mengambil keputusan. Dalam konsep dasar sistem basis data terdapat 4 (empat) komponen yang terdiri dari:

1. Data : Data didalam sebuah basis data dapat disimpan secara terintegrasi (*Integrated* dan data dapat dipakai secara bersama).
2. *Hardware* : *Hardware* terdiri dari semua peralatan komputer yang digunakan untuk pengelolaan sistem basis data.
3. *Software* : *Software* berfungsi segabai perantara (*interface*) antara pemakai dengan data fisik pada basis data.
4. *User* : *User* berfungsi sebagai yang mengakses basis data.

Kumpulan *file* yang saling berkaitan dan program untuk pengelolanya disebut DBMS. Bahasa yang terdapat di dalam *Database Management System*, yaitu:

Data Definition Language (DDL) atau memanipulasi data sebagai yang diorganisasikan sebelumnya model data yang tepat. DDL adalah pola *schema* basis data dispesifikasikan dengan satu set definisi yang diekspresikan dengan satu bahasa khusus.

Data Manipulation Language (DML) adalah bahasa yang memperbolehkan pemakai mengakses atau memanipulasi data. DML dapat mengambil informasi yang tersimpan dalam basisdata, menyisipkan informasi baru atau menghapus informasi dari basis data.

2.7 Web-service

World Wide Web Consortium (W3C) mendefinisikan *web* service sebagai sebuah *software* aplikasi yang dapat teridentifikasi oleh *URI* dan memiliki

interface yang didefiniskan, dideskripsikan, dan dimengerti oleh XML dan juga mendukung interaksi langsung dengan software aplikasi yang lain dengan menggunakan message berbasis XML melalui protokol internet.

Web service adalah sebuah *sofware* aplikasi yang tidak terpengaruh oleh *platform, web service* akan menyediakan *method-method* yang dapat diakses oleh *network*. W3C juga akan menggunakan XML untuk pertukaran data, khususnya pada dua *entities* bisnis yang berbeda.Beberapa karakteristik dari web service adalah:

1. *Message-based*
2. *Standards-based*
3. *Programming language independent*
4. *Platform-neutral*

Beberapa *key standard* didalam *web service* adalah: XML, SOAP, WSDL and UDDI.

2.8 Web Services Description Language

Web Services Description Language (WSDL) adalah sebuah *XML-based language* untuk mendeskripsikan XML. WSDL menyediakan *service* yang mendeskripsikan *service request* dengan menggunakan *protokol-protokol* yang berbeda dan juga *encoding*. WSDL akan memfasilitasi komunikasi antar aplikasi. WSDL akan mendeskripsikan apa yang akan dilakukan oleh web service, bagaimana menemukannya dan bagaimana untuk mengoperasikannya.

Spesifikasi WSDL mendefinisikan tujuh tipe element:

1. *Types* – element untuk mendefinisikan tipe data. Mereka akan mendefinisikan tipe data (seperti *string* atau *integer*) dari element didalam sebuah *message*.

2. *Message* - *abstract*, pendefinisian tipe data yang akan dikomunikasikan.
3. *Operation* – sebuah deskripsi *abstract* dari sebuah action yang didukung oleh *service*.
4. *Port Type* – sebuah koleksi *abstract* dari *operations* yang didukung oleh lebih dari satu *endpoints*.
5. *Binding* – mendefinisikan penyatuan dari tipe *port* (koleksi dari operasi-operasi) menjadi sebuah protokol transport dan data format (ex. SOAP 1.1 pada HTTP). Ini adalah sebuah *protocol* konkret dan sebuah spesifikasi data format didalam tipe *port* tertentu.
6. *Port* – mendefinisikan sebuah komunikasi endpoint sebagai kombinasi dari binding dan alamat network. Bagi protokol HTTP, ini adalah sebuah bentuk dari URL sedangkan bagi protokol SMTP, ini adalah sebuah *form* dari *email address.*
7. *Service* – satu set *port* yang terkorelasi atau suatu *endpoints*.

WSDL mendefinisikan *service* sebagai sebuah koleksi dari *endpoints network.* Sebuah definisi abstrak dari *endpoints* dan messages bersifat terpisah dari pembangunan *network* atau penyatuan data format. Pembagian ini menyebabkan penggunaan kembali *abstract description* dari data yang akan dipertukarkan (*message exchange*) dan *abstract collection* dari operasi (*ports*). *Protocol konkret* dan spesifikasi data format bagi tipe *port* tertentu menentukan *binding* yang dapat digunakan kembali (*reusable*). Sebuah port adalah sebuah *network address* yang dikombinasikan *reusable binding*; sebuah *service* adalah koleksi dari *port-port.*

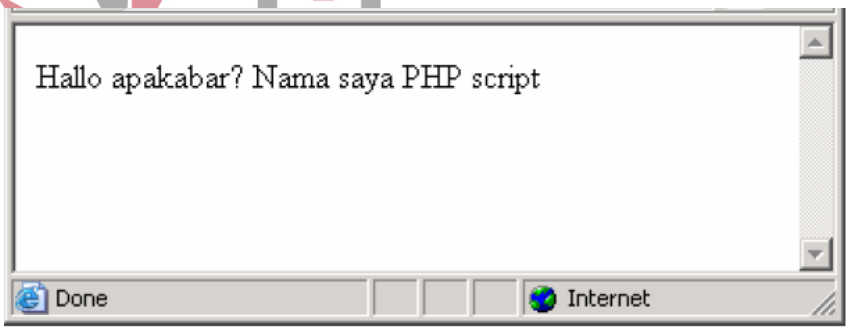
2.9 Hypertext Preprocessor

Hypertext Preprocessor (PHP) adalah bahasa scripting yang menyatu dengan HTML dan dijalankan pada *server side*. Artinya semua sintaks yang kita berikan akan sepenuhnya dijalankan pada server sedangkan yang dikirimkan ke browser hanya hasilnya saja.

```
<html>
<head>
      <title>
      Contoh Sederhana
      </title>
</head>
<body>
<?php
echo("Hallo apakabar? Nama saya PHP
script");
?>
</body>
</html>
```

Gambar 0.10 Contoh Script PHP

Maka hasil yang nantinya akan keluar adalah :



Gambar 0.11 Hasil Contoh Script PHP Saat Membuka Browser

Dalam PHP setiap nama variable diawali tanda dollar ($). Misalnya nama variable a dalam PHP ditulis dengan $a. Jenis suatu *variable* ditentukan pada saat jalannya program dan tergantung pada konteks yang digunakan. PHP mempunyai beberapa struktur kontrol, antara lain :

IF. Konstruksi *IF* digunakan untuk melakukan eksekusi suatu statement secara bersyarat. Cara penulisannya adalah sebagai berikut:

```
If (syarat)
{
      statement
}
```

Gambar 0.12 *Syntax* IF

While. Bentuk dasar dari statement *While* adalah sebagai berikut :

```
While (syarat)
{
      statement
}
```

Gambar 0.13 *Syntax While*

Arti dari statement *While* adalah memberikan perintah untuk menjalankan statement dibawahnya secara berulang-ulang, selama syaratnya terpenuhi.

FOR. Cara penulisan statement *FOR* adalah sebagai berikut:

```
For (ekspresi1; ekspresi2; ekspresi3)
{
      statement
}
```

Gambar 0.14 *Syntax* For

ekspresi1 menunjukkan nilai awal untuk suatu variable

ekspresi2 menunjukkan syarat yang harus terpenuhi untuk menjalankan statement

ekspresi3 menunjukkan pertambahan nilai untuk suatu variable.

SWITCH. *Statement SWITCH* digunakan untuk membandingkan suatu variable dengan beberapa nilai serta menjalankan statement tertentu jika nilai variable sama dengan nilai yang dibandingkan. Struktur *Switch* :

```
Switch (variable)
      case nilai1:
            statement;
            break;
      case nilai2:
            statement;
            break
      case nilai:
            statement;
      case else:
            break;
```

Gambar 0.15 *Syntax switch*

REQUIRE. Statement *Require* digunakan untuk membaca nilai variable dan fungsi-fungsi dari sebuah file lain. Cara penulisan statement *Require* adalah :

```
Require (nama file);
```

Statement *Require* ini tidak dapat dimasukkan diadalam suatu struktur looping misalnya *While* atau *For*. Karena hanya memperbolehkan pemangggilan file yang sama tersebut hanya sekali saja.

File contoh9.php:

```
<?php
$a="Saya sedang belajar PHP";
function tulistebal($teks)
{
echo("<b>$teks</b>");
}
?>
<?php
Require("contoh9.php");
tulistebal("Ini adalah tulisan tebal");
echo("<br>");
echo($a);
?>
```

Gambar 0.16 Contoh *source code* php

Hasilnya adalah:

Gambar 0.17 Contoh hasil script *Require*

INCLUDE. Statement *Include* akan menyertakan isi suatu file tertentu. *Include* dapat diletakkan didalam suatu looping misalkan dalam statement *For* atau *While*.

File contoh11.php:

```
<?php
echo("----------------------------------------<br>");
echo("PHP adalah bahasa scripting<br>");
echo("----------------------------------------<br>");
echo("<br>");
?>
<?php
For ($b=1; $b<5; $b++)
{
Include("contoh11.php");
}
?>
<?php
For ($b=1; $b<5; $b++)
{
Include("contoh11.php");
}?>
```

Gambar 0.18 *Source Code* PHP

Hasilnya adalah:

Gambar 0.19 Contoh hasil script statement *INCLUDE*

2.10 Mobile Web

Mobile web adalah aplikasi akses internet atau akses internet menggunakan peralatan yang bersifat *mobile* berbasiskan *browser* yang bertujuan untuk mengakses layanan data secara *wireless* dengan menggunakan perangkat *mobile* seperti *handphone, smartphone,* PDA dan perangkat *portable* yang tersambung ke sebuah jaringan telekomunikasi selular.

Mobile web yang diakses melalui perangkat *mobile* perlu dirancang dengan mempertimbangkan keterbatasan perangkat *mobile* seperti sebuah *handphone* yang memiliki layar dengan ukuran yang terbatas ataupun beberapa keterbatasan pada sebuah perangkat *mobile*

2.11 Simple Object Access Protocol

Simple Object Access Protocol (SOAP) adalah sebuah XML-*based mark-up language* untuk pergantian pesan diantara aplikasi-aplikasi. SOAP berguna seperti sebuah amplop yang digunakan untuk pertukaran data *object* didalam *network*. SOAP mendefinisikan empat aspek didalam komunikasi: *Message envelope, Encoding, RPC call convention,* dan bagaimana menyatukan sebuah *message* didalam protokol transport. Sebuah SOAP *message* terdiri dari SOAP *Envelope* dan bisa terdiri dari *attachments* atau tidak memiliki *attachment*. SOAP *envelope* tersusun dari SOAP header dan SOAP *body*, sedangkan SOAP *attachment* membolehkan non-XML data untuk dimasukkan kedalam SOAP message, di-*encoded,* dan diletakkan kedalam SOAP *message* dengan menggunakan MIME-multipart.

2.12 Visual Basic .NET

Microsoft Visual Basic .NET adalah sebuah alat untuk mengembangkan dan membangun aplikasi yang bergerak di atas sistem .NET *Framework*, dengan menggunakan bahasa BASIC. Dengan menggunakan alat ini, para pembuat program dapat membangun aplikasi Windows *Forms*. Alat ini dapat diperoleh secara terpisah dari beberapa produk lainnya (seperti Microsoft Visual C++, Visual C#, atau Visual J#), atau juga dapat diperoleh secara terpadu dalam Microsoft Visual Studio .NET. Bahasa Visual Basic .NET sendiri menganut paradigma bahasa pemrograman berorientasi objek yang dapat dilihat sebagai evolusi dari Microsoft Visual Basic versi sebelumnya yang diimplementasikan di atas .NET Framework. Peluncurannya mengundang kontroversi, mengingat banyak sekali perubahan yang dilakukan oleh Microsoft, dan versi baru ini tidak kompatibel dengan versi terdahulu.

Microsoft .NET Framework (dibaca Microsoft Dot Net Framework) adalah sebuah komponen yang dapat ditambahkan ke sistem operasi Microsoft Windows atau telah terintegrasi ke dalam Windows (mulai dari Windows Server 2003 dan versi-versi Windows terbaru). Kerangka kerja ini menyediakan sejumlah besar solusi-solusi program untuk memenuhi kebutuhan-kebutuhan umum suatu program baru, dan mengatur eksekusi program-program yang ditulis secara khusus untuk *framework* ini. .NET Framework adalah kunci penawaran utama dari *Microsoft*, dan dimaksudkan untuk digunakan oleh sebagian besar aplikasi-aplikasi baru yang dibuat untuk *platform Windows.*

Pada dasarnya, .NET Framework memiliki 2 komponen utama: CLR dan .NET Framework *Class Library*. Program - program yang ditulis untuk .NET

Framework dijalankan pada suatu lingkungan software yang mengatur persyaratan-persyaratan *runtime* program. *Runtime environment* ini, yang juga merupakan suatu bagian dari .NET Framework, dikenal sebagai Common Language Runtime (CLR). CLR menyediakan penampilan dari *application virtual machine*, sehingga para *programmer* tidak perlu mengetahui kemampuan CPU tertentu yang akan menjalankan program. CLR juga menyediakan layanan-layanan penting lainnya seperti jaminan keamanan, pengaturan *memori*, *garbage collection* dan *exception handling* / penanganan kesalahan pada saat *runtime*. *Class library* dan CLR ini merupakan komponen inti dari .NET Framework.

Kerangka kerja itu pun dibuat sedemikian rupa agar para programmer dapat mengembangkan program komputer dengan jauh lebih mudah, dan juga untuk mengurangi kerawanan aplikasi dan juga komputer dari beberapa ancaman keamanan.

CLR adalah turunan dari CLI (*Common Language Infrastructure*) yang saat ini merupakan standar ECMA. Untuk keterangan lebih lanjut, silakan mengunjungi situs ECMA atau kunjungi sumber pranala di bawah artikel ini.

Solusi-solusi program pembentuk class *library* dari .NET Framework melakukan proses *cover* area yang luas dari kebutuhan program pada bidang *user interface*, pengaksesan data, koneksi basis data, kriptografi, pembuatan aplikasi berbasis web, algoritma numerik, dan komunikasi jaringan. Fungsi-fungsi yang ada dalam *class library* dapat digabungkan oleh *programmer* dengan kodenya sendiri untuk membuat suatu program aplikasi baru.

2.13 MySQL

MySQL adalah sebuah program database server yng mampu menerima dan mengirimkan datanya sangat cepat, multi user serta menggunakan perintah dasar SQL (*Structured Query Language*). MySQL merupakan dua bentuk lisensi, yaitu *freeSoftware* dan *Shareware*. MySQL yng biasa kita gunakan adalah mySQL *FreeSoftware* yang berada di bawah Lisensi GNU/GPL (*General Public License*).

MySQL merupakan sebuah *database server* yang *free*, artinya kita bebas menggunakan *database* ini untuk keperluan pribadi atau usaha tanpa harus membeli atau membayar lisensinya. MySQL pertama kali dirintis oleh seorang *programme database* bernama Michael Widenius. Selain *database server*, mySQL juga merupakan program yang dapat mengakses suatu database mySQL yang berposisi sebagai server, yang berarti program kita berposisi sebagai *Client*. Jadi MySQL adalah sebuah *database* yang dapat digunakan sebagai *Client* maupun *server*.

Database mySQL merupakan suatu perangkat lunak *database* yang berbentuk *database relational Database Management System* (*RDBMS*) yang menggunakan suatu bahasa permintaan yang bernama SQL (*Structured Query Language*). Kelebihan MySQL *Database* MySQL memiliki beberapa kelebihan dibanding *database* lain, diantaranya :

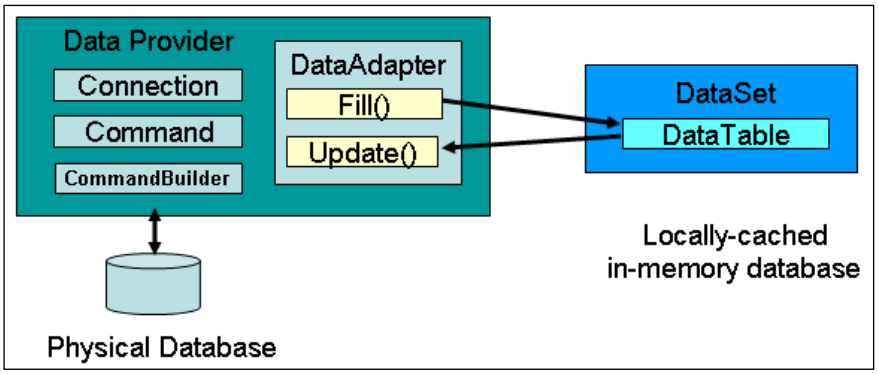
1. MySQL merupakan *Database Management System* (DBMS)
2. MySQL sebagai *Relation* Database *Managemet System* (*RDBMS*) atau disebut dengan *database Relational*

3. MySQL merupakan sebuah *database server* yang *free*, artinya kita bebas menggunakan database ini untuk keperluan pribadi atau tanpa usaha tanpa harus membeli atau membayar lisensinya.
4. MySQL merupakan sebuah *database client*.
5. MySQL mampu menerima *query* yang bertumpuk dalam satu permintaan atau *Multi-Threading*
6. MySQL merupakan d*atabase* yang mampu menyimpan data berkapasitas sangat besar hingga berukuran *gigabyte* sekalipun.
7. MySQL didukung oleh *driver* ODBC, artinya *database* MySQL dapat diakses menggunakan aplikasi apa saja termasuk berupa *visual* seperti visual Basic dan Delphi.
8. MySQL adalah data*b*ase yang menggunakan enkripsi *password*, jadi *database* ini cukup aman karena memiliki *password* untuk mengaksesnya.
9. MySQL merupakan database *server* yang *multi user*, artinya *database* ini tidak hanya digunakan oleh satu pihak orang akan tetapi dapat digunakan oleh banyak pengguna.
10. MySQL mendukung field yang dijadikan sebagai kunci primer dan kunci unik (*Unique*)
11. MySQL memiliki kecepatan dalam pembuatan table maupun peng-*update* an table.

2.14 Disconnected Database Model

Disconnected database model merupakan salah satu teknik aplikasi untuk membuat koneksi ke database yang cukup lama supaya dapat digunakan untuk mengambil *snapshot* dari data lalu memisahkannya. Pada akhirnya dapat

digunakan untuk melakukan fungsi *add/delete/modify* pada data yang sudah terkoneksi melalui koneksi lokal pada PC.



Gambar 0.20 *Disconnected database object model*

```
Dim con As New OleDbConnection()
Dim cmd As New OleDbCommand()
Dim da As New OleDbDataAdapter()
Dim cb As OleDbCommandBuilder
Dim dr As DataRow
Dim dt As New DataTable()

' a Connection object to locate our database
con.ConnectionString    =    "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=..\test.mdb"

' a Command object to get the data
cmd.Connection = con
cmd.CommandText = "Select * from Table1"

' a DataAdpter object to fill the data
da.SelectCommand = cmd
da.Fill(dt)

' let's add a record
dr = dt.NewRow()
dr("LName") = "Doe"
dr("FName") = "John"
dr("IDNo") = 120
dt.Rows.Add(dr)

' let's delete a record
dr = dt.Rows(2)
dr.Delete()

' let's edit the contents of a record
dr = dt.Rows(1)
dr("ID") = -120

' Create the SQL insert/delete/update commands that will be used
by the data adapter's
' update method below. This is optional... you *could* populate
the 3 properties of the
' data adapter yourself (but why?)
```

```
cb = New OleDbCommandBuilder(da)
Try
    ' Now let's write the changes back to the database
     da.Update(dt)
Catch ex As Exception
     MsgBox("Yikes, Can't update the database" & vbCr & ex.Message, _
      MsgBoxStyle.Critical, "Error!")
     Exit Sub
End Try
```

Gambar 0.21 Contoh kode pada *disconnected Database Model*

Jika ingin memanggil ulang *database* yang akan digunakan, dapat digunakan perintah metode *DataAdapter's Fill*() untuk membaca konten dari tabel yang ada didalam database dan mengisi *DataTable* objek yang terletak pada *cache* lokal. Metode yang lain adalah metode *update*(). Metode *update*() biasanya melakukan proses *scan* pada *DataTable* yang terletak di koneksi lokal untuk mengambil beberapa baris yang telah dimodifikasi. Ada beberapa *shortcut method* dari penulisan tiga perintah SQL yang biasa disebut *CommandBuilder*.


INSTITUT BISNIS
& INFORMATIKA
stikom
SURABAYA