2. TEORI PENUNJANG

2.1 *Hypertext Markup Language*

Hypertext Markup Language (HTML) adalah bahasa yang merupakan *plaintext* (lebih dikenal sebagai ASCII) yang digunakan untuk membangun suatu halaman web dan dapat dibuat menggunakan beberapa *text editor* seperti Emacs atau Vi pada UNIX, simple text pada Macintosh atau Notepad pada Windows. HTML merupakan dasar dari pemrograman web. Dengan HTML, data berupa teks, gambar, suara dan link dapat digabungkan menjadi satu.

Halaman web yang dibuat dengan skrip HTML murni, tanpa ditambah skrip lain akan bersifat statis. Halaman web tersebut hanya dapat dibaca tetapi tidak dapat diubah maupun dieksekusi oleh *user* lain. Selain itu, HTML juga memiliki sifat fleksibel karena dapat dikombinasikan dengan skrip atau bahasa pemrograman lainnya.

HTML menggunakan *tag* dalam penulisan programnya. Tag adalah penandaan dalam HTML. Tag selalu ditulis di antara tanda lebih kecil (<) dan tanda lebih besar (>), seperti berikut:

```
<tag>
Struktur file HTML harus mempunyai:
<HTML>
……..
</HTML>
```

Penulisan tag HTML tidak bersifat *case sensitive*. Maksudnya bahwa penulisan tag dengan huruf kapital hanya untuk mempermudah perbedaan antara teks biasa dengan tag.

Secara sederhana HTML terdiri dari dua bagian yaitu *header* dan *body*. Struktur HTML diapit oleh tag awal <HTML> dan tag akhir </HTML>. Standar penulisannya:

```
<HTML>
<HEAD>
        Penjelasan Dokumen
</HEAD>
<BODY>
        Isi Dokumen
</BODY>
</HTML>
```

Bagian *head* biasanya berisi informasi mengenai dokumen tersebut, misalnya judul dokumen, versi HTML yang digunakan, dan lain-lain. Sedangkan *body* berisi desain halaman web.

2.1.1. *Heading*

Heading adalah sekumpulan kata yang menjadi judul atau subjudul dalam suatu dokumen HTML. HTML menyediakan enam tingkatan *heading*. *Heading* level 1 biasanya untuk judul utama. Untuk lebih jelasnya ditampilkan pada sontoh berikut:

```
<HTML>
        <HEAD>
                <TITLE> Heading </TITLE>
        </HEAD>
        <BODY>
                <H1> Heading level 1 </H1>
<H2> Heading level 2 </H2>
<H3> Heading level 3 </H3>
<H4> Heading level 4 </H4>
<H5> Heading level 5 </H5>
<H6> Heading level 6 </H6>
        </BODY>
</HTML>
```

2.2. PHP

PHP yang merupakan singkatan dari *PHP Hypertext Prepocessor* adalah suatu bahasa pemrograman yang bersifat *server-side* dan didesain khusus untuk web. Kode PHP dapat diimplementasikan pada halaman HTML yang akan dieksekusi setiap kali halaman tersebut dikunjungi. Kode PHP akan diinterpretasikan ke dalam *web server* dan dijalankan pada HTML sehingga pengunjung dapat melihat hasilnya.

2.2.1. Sejarah PHP

PHP diciptakan oleh Rasmus Lerdorf. Semula PHP hanya digunakan untuk mencatat jumlah pengunjung pada *homepage* yang dibuat oleh Rasmus. Kemudian Rasmus mengeluarkan *Personal Homepage Tools* versi 1.0 secara gratis. Secara khusus PHP dirancang untuk membentuk web dinamis, artinya dapat membentuk suatu tampilan yang berdasarkan permintaan terkini, misalnya tampilan *database* pada halaman web.

Pada tahun 1995, Rasmus menciptakan PHP versi 2.0. Pada versi inilah *programmer* dapat menempelkan kode terstruktur dalam tag HTML. Selain itu kode PHP juga bisa berkomunikasi dengan *database* dan melakukan perhitungan-perhitungan yang kompleks sambil jalan.

Pada tahun 1998 PHP versi 3.0 resmi dikeluarkan. Kemudian pada tahun 2000 PHP versi 4.0 diterbitkan. Pada saat ini PHP cukup populer sebagai perangkat pemrograman web, terutama di lingkungan Linux. Namun PHP juga dapat berfungsi pada *server* yang berbasis UNIX, Windows NT, dan Macintosh. Berdasarkan hasil survei dari Netcraft (http://www.netcraft.com), PHP adalah salah satu bahasa yang populer.

2.2.2. Kelebihan-kelebihan PHP

Pada awalnya orang cukup puas dengan situs statis, namun seiring dengan semakin berkembangnya *e-commerce*, maka kebutuhan akan situs dinamis yang berjalan 24 jam sehari semakin meningkat. Dalam hal ini PHP memiliki beberapa kelebihan dibanding dengan bahasa pemrograman yang lain.

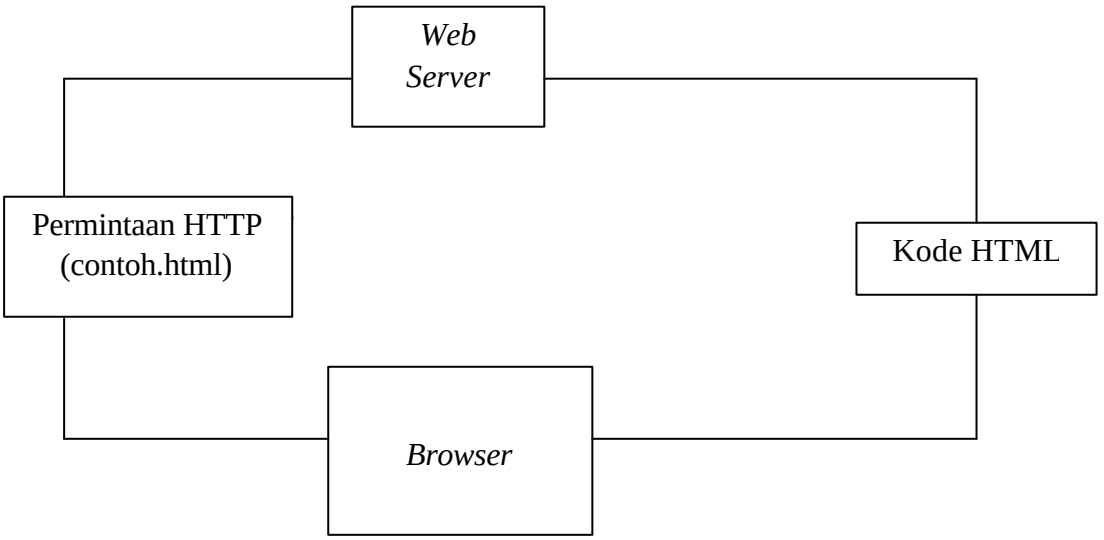
PHP mudah dibuat dan cepat dijalankan. PHP dapat berjalan dalam web server yang berbeda dan dalam sistem operasi yang berbeda pula. PHP bersifat *open source*, maksudnya adalah *source code* PHP dapat digunakan dan diedit tanpa harus mengeluarkan biaya. Tidak perlu membayar apapun untuk menggunakan perangkat lunak ini, karena dapat di *download* melalui situs www.php.net. Untuk versi Windows dapat diperoleh kode binernya, dan untuk versi Linux dapat diperoleh kode sumbernya secara lengkap.

PHP juga dapat berjalan pada *Microsoft Personal Web Server*, Apache, IIS dan sebagainya. PHP yang ditulis dengan bahasa C dapat dikembangkan sendiri, sehingga bagi yang menguasai bahasa C dapat dengan mudah menambahkan fungsi-fungsi baru.

2.2.3. Konsep Kerja PHP

Model kerja HTML diawali dengan permintaan suatu halaman web oleh *browser*. Berdasarkan *Uniform Resosurce Locator* (URL) atau dikenal dengan sebutan alamat Internet, *browser* mendapatkan alamat dari *web server*, mengidentifikasi halaman yang dikehendaki, dan menyampaikan segala informasi yang dibutuhkan oleh *web server*. Informasi yang disampaikan ke *web server* antara lain nama *browser,* versinya, dan sistem operasinya.

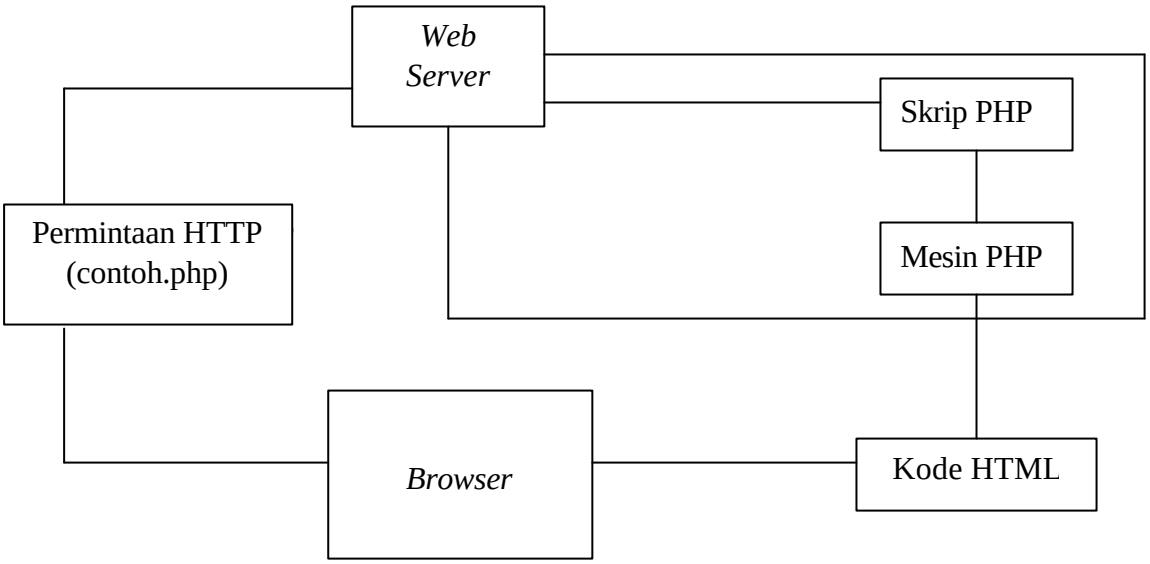
Selanjutnya *web server* akan mencarikan berkas yang diminta dan memberikan isisnya ke *browser*. *Browser* yang mendapatkan isinya segera melakukan proses penerjemahan kode HTML dan menampilkannya ke layar pemakai. Untuk lebih jelasnya, lihat Gambar 2.1.



Gambar 2.1 Skema HTML

Apabila yang diminta adalah sebuah halaman PHP, maka prinsipnya serupa dengan kode HTML. Hanya saja, ketika berkas PHP yang diminta didapatkan oleh *web server*, isinya segera dikirimkan ke mesin PHP dan mesin

inilah yang memproses dan memberikan hasilnya (berupa kode HTML) ke web server. Selanjutnya web server menyampaikan ke klien.



Gambar 2.2 Skema PHP

2.2.4. *Script* PHP

Skrip PHP berkedudukan sebagai tag dalam bahasa HTML. Perintah PHP ditandai dengan tag pembuka <? Dan tag penutup ?>. Ada beberapa cara penulisan tag PHP antara lain:

a.) Cara 1, merupakan cara yang paling ringkas dan paling umum digunakan.

```
<?
  script PHP
?>
```

b.) Cara 2 digunakan apabila *script* PHP dikombinasikan dengan *script* XML.

```
<?php
  script PHP
?>
```

c.) Cara 3 digunakan untuk mengantisipasi teks editor yang tidak dapat menerima penulisan dengan cara 1 maupun cara 2.

```
<SCRIPT LANGUAGE = "PHP">
  script PHP
</SCRIPT>
```

Berikut ini merupakan contoh *script* PHP:

```
<HTML>
<HEAD>
<TITLE> Menggunakan PHP </TITLE>
</HEAD>
<BODY>
<?
$nama = "User";
Printf("Halo, %s", $nama);
?>
</BODY>
</HTML>
```

Pada *script* di atas, "nama" adalah nama variabel. Pada program PHP, variabel selalu ditulis dengan diawali tanda $. Baris:

$nama = "User";

Merupakan pernyataan yang digunakan untuk memberikan string "User" ke variabel "nama". Tanda "%s" merupakan tanda format untuk string (artinya, data yang akan mensubsitusikan tanda tersebut adalah data string). Jika *script* di atas dijalankan maka hasilnya:

Halo, User

2.2.5. Perintah – perintah PHP

Ada beberapa perintah yang sering digunakan dalam bahasa pemrograman PHP, antara lain:

a.) IF

Perintah ini digunakan untuk menjalankan satu atau lebih perintah yang menyatakan keadaan. Standar penulisannya adalah:

```
If (kondisi)
        {
                Pernyataan akan dijalankan jika kondisi bernilai benar
        }
```

b.) IF…ELSE

Perintah jenis ini mirip dengan perintah sebelumnya, hanya saja digunakan untuk banyak blok perintah. Standar penulisannya sebagai berikut:

```
If (kondisi)
        {
                Pernyataan 1 akan dijalankan jika kondisi benar
        }
ElseIf (kondisi 2)
        {
Pernyataan 2 akan dijalankan jika kondisi 2 bernilai benar dan kondisi 1 bernilai
salah
                }
        Else
                {
Pernyataan ini akan dijalankan jika kondisi 1 dan 2 bernilai salah
                }
```

PHP pertama kali akan menguji kondisi pertama, jika bernilai salah maka akan terus diuji ke kondisi berikutnya sampai ditemukan kondisi yang benar. Jika kondisi yang benar sudah ditemukan, maka akan dijalankan pernyataan yang ada dalam kondisi tersebut.

c.) FOR

Perintah FOR digunakan untuk mengulangi perintah dengan jumlah pengulangan yang sudah diketahui. Pada perintah ini tidak perlu menuliskan sebuah kondisi untuk diuji. Jadi hanya perlu menuliskan nilai awal dan nilai akhir variabel penghitung. Nilai variabel akan otomatis bertambah atau berkurang tiap kali sebuah pengulangan diselesaikan. Standar penulisannya:

```
For (nilai awal, nilai akhir, peningkatan/penurunan
{
        Pernyataan yang akan dijalankan
}
```

d.) WHILE

Perintah ini digunakan untuk mengulangi sebuah perintah sampai jumlah tertentu. Untuk menghentikan pengulangan digunakan suatu kondisi tertentu. Nilai kondisi ini, seperti halnya pada perintah IF…ELSE, mempunyai hasil akhir berupa salah (*False*) atau benar (*True*). Pengulangan akan terus berjalan selama kondisi masih bernilai benar. Adapun penulisannya:

e.) DO…WHILE

```
While(kondisi)
{
        Pernyataan yang akan dijalankan
}
```

Perintah ini mirip perintah while. Proses pengulangan akan berjalan jika kondisi yang diperiksa di while masih bernilai benar dan pengulangan akan dihentikan jika kondisisnya sudah bernilai salah. Penulisan untuk perintah ini:

```
Do
{
        Pernyataan yang akan dijalankan
}
While(kondisi)
```

2.2.6. Operator-operator dalam PHP

Operator adalah simbol yang digunakan dalam program untuk melakukan suatu operasi, misalnya penjumlahan atau perkalian, pembandingan kesamaan dua buah nilai, atau bahkan memberikan nilai ke variabel. Nilai yang dioperasikan oleh operator (disebut *operand* atau argumen) bersama-sama operator membentuk ekspresi (ungkapan). Contoh:

```
1+2*3
```

disebut ekspresi. Tanda + dan * disebut operator, sedangkan 1,2, dan 3 adalah *operand*. Pada PHP ada beberapa operator yang biasa digunakan, antara lain:

a.) Operator Aritmetika

Operator ini digunakan dalam operasi matematika. Adapun beberapa operator aritmetika pada PHP, dapat dilihat pada table 2.1.

Tabel 2.1. Operator-operator Aritmetika

Operator	Kegunaan	Prioritas
+	Penjumlahan	Ketiga
-	Pengurangan	Ketiga
*	Perkalian	Kedua
/	Pembagian	Kedua
%	Sisa Pembagian	Kedua
++	Penaikan	Pertama
--	Penurunan	Pertama

b.) Operator Penugasan

Operator penugasan digunakan untuk memberikan nilai ke suatu variabel. Salah satu operator penugasan yang sering digunakan adalah '='. Beberapa operator penugasan dapat dilihat pada tabel 2.2.

Tabel 2.2. Operator-operator Penugasan

Operator	Kegunaan	Contoh
+=	Menambahkan variabel di sisi kiri dengan nilai di sisi kanan	X += 2; identik dengan: X :=X+2;
-=	Mengurangi isi variabel di sisi kiri dengan nilai di sisi kanan	X -= 2; identik dengan: X :=X-2;
/=	Membagi variabel di sisi kiri dengan nilai di sisi kanan	X /= 2; identik dengan: X := X/2;
%=	Memperoleh sisa pembagian antara variabel di sisi kiri dengan nilai di sisi kanan	X %=2 ; identik dengan: X := X % 2;
&=	Melakukan operasi "dan" / "and" terhadap variabel di sisi kiri dengan nilai di sisi kanan	X &= 2; identik dengan: X := X & 2;
|=	Melakukan operasi "atau" / "or" terhadap variabel di sisi kiri dengan nilai di sisi kanan	X |=2; identik dengan: X := X | 2;
^=	Melakukan operasi "atau ekslusif" / "xor" terhadap variabel di sisi kiri dengan nilai di sisi kanan	X ^ = 2; identik dengan: X := X ^ 2;
:=	Melakukan operasi konkatenasi terhadap variabel di sisi kiri dengan nilai di sisi kanan	X := 'A'; identik dengan: X := X.'A';

c.) Operator Perbandingan

Operator perbandingan digunakan untuk melakukan perbandingan antara dua *operand* dan menghasilkan nilai benar atau salah. Beberapa operator-operator perbandingan dapat dilihat pada tabel 2.3.

Tabel 2.3. Operator-operator Perbandingan

Operator	Makna
==	Sama dengan
<	Kurang dari
>	Lebih dari
<=	Kurang dari atau sama dengan
>=	Lebih dari atau sama dengan
!=	Tidak sama dengan
<>	Tidak sama dengan

d.) Urutan Pengerjaan Operator PHP

Bila sebuah ekspresi melibatkan lebih dari sebuah operator, masalah prioritas pada operator perlu diperhatikan karena menentukan urutan pengerjaan operator dalam ekspresi. Misalnya:

```
5 + 2 * 3
```

Maka hasilnya adalah 11 bukan 30. Persoalan seperti ini dapat dijawab dengan benar jika kita mengetahui prioritas dari operator-operator yang digunakan. Tabel berikut ini menunjukkan prioritas dari operator-operator pada PHP.

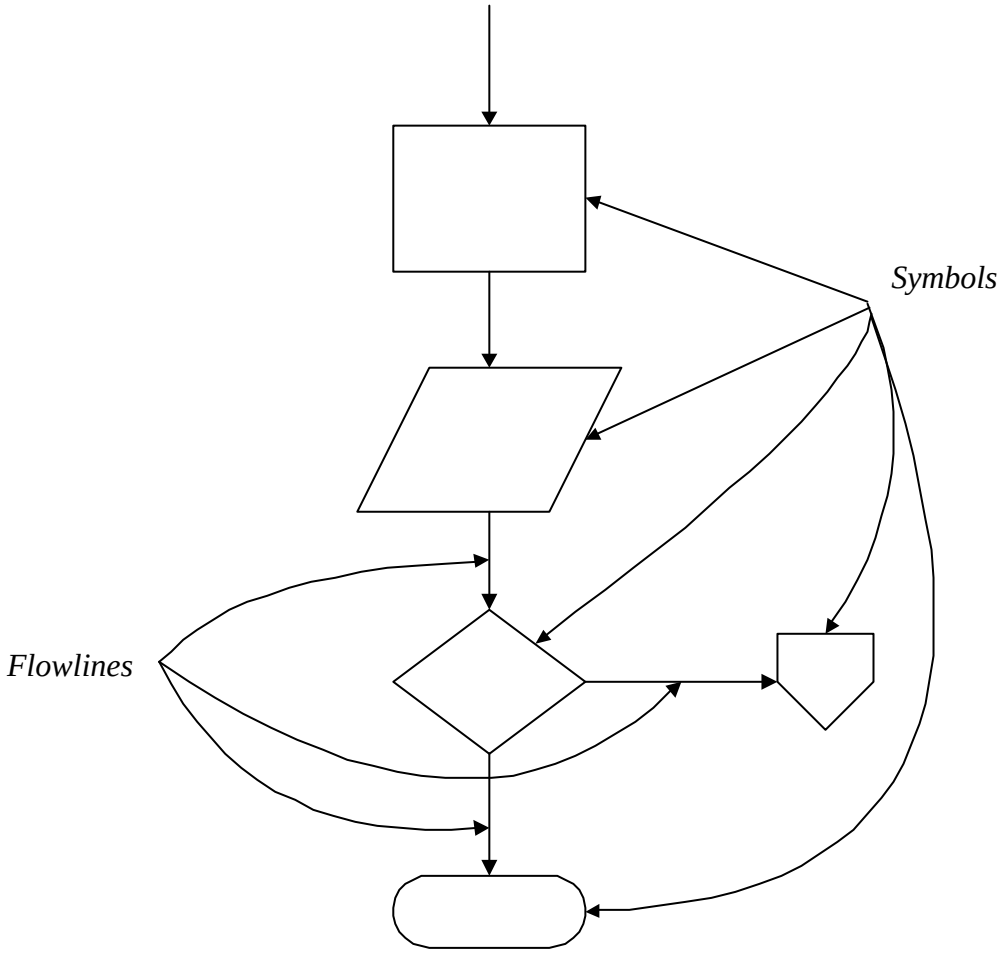
Tabel 2.4. Prioritas Operator pada PHP

Prioritas	Operator
Tertinggi	( ) { }
	~ ! ++ -- - $ &
	* / %
	+ -
	< > <= >=
	== !=
	&
	^
	!
	&&
	||
	= += -= /= &= |= ^= :=
	AND (&&)
	XOR (||)
Terendah	OR

2.3 *Flowchart*

Flowchart merupakan sebuah *tool* yang membantu *programmer* dalam membuat suatu program. Layaknya seperti seorang arsitek, maka sebelum program dibuat, terlebih dahulu diGambar sketsa programnya untuk memudahkan dalam pembuatan program. *Flowchart* mengandung langkah-langkah yang saling terintegrasi dalam suatu program.

Flowchart terdiri dari *symbol* yang menampilkan fungsi-fungsi dari program, dan *flowlines* yang merupakan alur yang menunjukkan proses atau langkah sesuai dengan pilihan dalam *flowchart*. Instruksi-instruksi yang dikerjakan ditulis ke dalam *symbol*. Gambar 2.3 merupakan Gambar *symbol* dan *flowlines.*



Gambar 2.3. *Symbol* dan *Flowlines*

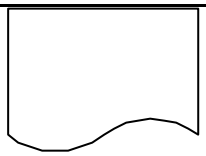
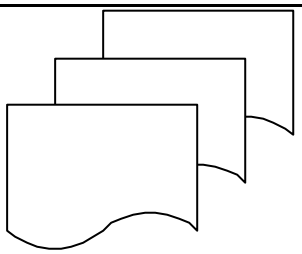
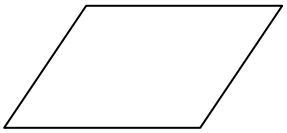
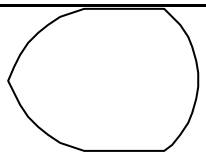
Program *flowchart* dibaca dari atas ke bawah, sesuai dengan aliran *flowlines.* Selain itu juga dapat dibaca secara vertikal sesuai dengan pilihan dalam alur tersebut.

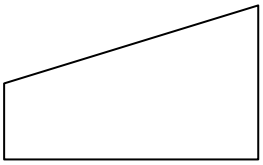
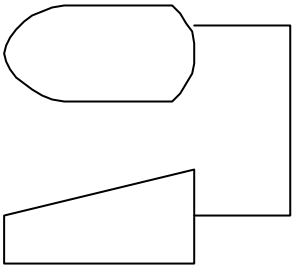
2.3.1. Simbol-simbol *Flowchart*

Ada beberapa simbol yang sering digunakan dalam pembuatan suatu *flowchart*. Simbol-simbol tersebut mempunyai pengertian yang berbeda-beda. Simbol yang digunakan untuk membuat *flowchart* terbagi atas 4 macam yaitu:

a.) *Input/output symbol*

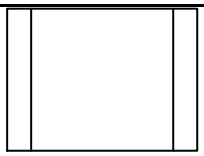
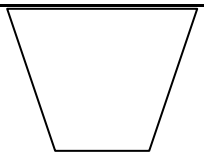
Tabel 2.5. *Input/output Symbols*

Simbol	Nama	Keterangan
	*Document*	Sebuah dokumen atau laporan secara manual ataupun menggunkan komputer
	*Multi Document*	Beberapa salinan dari sebuah dokumen, yang dilambangkan dengan dokumen yang berlapis-lapis.
	*Input/Output*	Menunjukkan sebuah masukan/keluaran untuk suatu proses yang diGambarkan dalam *flowchart*. Juga dapat melambangkan jurnal dan pembukuan perusahaan
	*Display*	Info yang ditampilkan dengan sebuah alat elektronik yang terhubung secara *online*, contohnya komputer (PC)

	*Input/Output* secara *online*	Proses memasukkan data dengan menggunakan alat elektronik yang terhubung secara *online*
	Terminal CRT, komputer (PC)	Simbol layar tampilan dan simbol input data secara *online* digunakan bersama untuk menunjukkan sebuah terminal CRT atau komputer (PC)

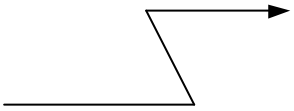
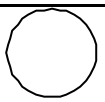
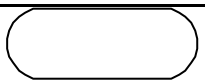
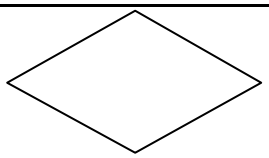
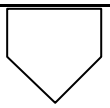
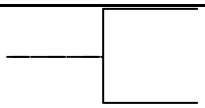
b.) *Processing Symbol*

Tabel 2.6. *Processing Symbols*

Simbol	**Nama**	**Keterangan**
	*Predefined Process* (secara manual)	Fungsi pemrosesan menggunakan komputer, biasanya ditandai dengan perubahan data/informasi
	Operasi Manual	Fungsi pemrosesan secara manual (tidak menggunakan alat)
	Operasi Tambahan	Fungsi pemrosesan oleh suatu alat selain komputer

c.) *Flow and Miscellenous Symbol*

Tabel 2.7 *Flow and Miscellenous Symbols*

Simbol	Nama	Keterangan
	Line connector (Aliran dokumen/pemrosesan)	Aliran normal arahnya ke kanan/ke bawah
	Aliran data/informasi	Arah data/aliran informasi, biasanya menunjukkan data yang disalin dari suatu dokumen ke dokumen lainnya
	Penghubung komunikasi	Transmisi data dari suatu lokasi ke lokasi lainnya
	*Connector* dalam halaman yang sama	Menghubungkan aliran proses yang terputus pada halaman yang sama
	Terminal	Suatu awal, akhir, atau titik interupsi dalam suatu proses.
	*Decision*	Digunakan untuk mengambil keputusan dalam perbandingan logika (biasanya dalam bentuk *Yes or No question)*
	*Off page reference*	Masukan dari halaman sebelumnya, atau keluar ke halaman lain
	Catatan/keterangan (*annotation*)	Penambahan deskripsi perintah atau penjelasan untuk klarifikasi

2.3.2. Beberapa Anjuran dalam membuat program *Flowchart*

Untuk membuat suatu program *flowchart*, maka ada beberapa hal yang perlu diperhatikan, antara lain:

d.) Buat jalan proses yang sesingkat-singkatnya, hindari pengulangan proses yang tidak perlu dan logika yang berbelit-belit.

e.) Rangkaian-rangkaian proses yang sama sedapat mungkin diGambarkan satu kali saja.

f.) Gambarkan jalannya proses dari atas ke bawah atau dari kiri ke kanan.

g.) Sebuah program *flowchart* harus diawali dari satu titik 'START' dan diakhiri dengan 'STOP' atau 'END'.

2.4 MySQL

MySQL adalah *Relational Database Management System* (RDBMS) yang didistribusikan secara gratis dibawah lisensi *General Public License* (GPL). Setiap orang bebas menggunakan MySQL, namun tidak boleh dijadikan produk turunan yang bersifat *closed source* atau komersial.

MySQL merupakan turunan dari salah satu konsep utama dalam database yaitu *Structured Query Language* (SQL). SQL adalah sebuah konsep pengoperasian database terutama untuk pemilihan dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis.

Sebagai *database server*, MySQL termasuk unggul dibandingkan *database server* lainnya dalam query data. Hal ini dapat dibuktikan melalui kecepatan MySQL yang bisa sepuluh kali lebih cepat dari PostgreSQL dan lima kali lebih cepat dibanding Interbase.

2.4.1. Sejarah MySQL

MySQL dikembangkan sekitar tahun 1994 oleh sebuah perusahaan pengembang software dan konsultan *database* bernama MySQL AB yang berasal dari Swedia. Awalnya perusahaan tersebut bernama TcX DataConsult AB kemudian namanya berubah seiring dengan dikembangkannya MySQL yang berfungsi untuk mengembangkan aplikasi berbasis web pada *client*.

MySQL dikembangkan oleh Michael Widenius "Monty", yang pada awalnya merupakan pengembang satu-satunya TcX. Monty memiliki aplikasi UNIREG dan ISAM buatannya sendiri, kemudian ia mencari interface SQL yang cocok untuk diimplementasikan ke dalamnya. Semula Monty menggunakan miniSQL (mSQL) pada percobaannya itu, namun mSQL dirasa kurang sesuai karena terlalu lambat dalam pemrosesan query. Akhirnya Monty menghubungi David Hughes yang merupakan pembuat mSQL. Lalu Monty membuat sendiri mesin SQL yang mempunyai interface mirip dengan SQL, tetapi dengan kemampuan yang lebih sesuai, maka lahirlah MySQL.

Tentang pengambilan nama MySQL sampai saat ini masih belum jelas asal usulnya. Ada yang berpendapat bahwa nama *My* diambil dari huruf depan dan belakang Monty, tetapi ada juga yang mengatakan bahwa nama itu diambil dari nama putri Monty yang kebetulan juga bernama *My*.

2.4.2. Keistimewaan MySQL

MySQL merupakan *database server* yang memiliki konsep *database modern*. MySQL mempunyai beberapa keistimewaan sebagai berikut:

a.) Portability

MySQL dapat berjalan stabil pada berbagai sistem operasi di antaranya adalah seperti Windows, Linux, FreeBSD, Mac OS X Server, Solaris, Amiga, HP-UX, dan lain-lain.

b.) Open Source

MySQL didistribusikan secara *open source* (gratis), dibawah lisensi GPL sehingga dapat digunakan tanpa dipungut biaya sepeser pun.

c.) Multiuser

MySQL dapat digunakan oleh beberapa *user* dalam waktu yang bersamaan tanpa mengalami masalah atau konflik. Hal ini memungkinkan sebuah *database server* MySQL dapat diakses *client* secara bersamaan.

d.) Performance Tuning

MySQL memiliki kecepatan yang baik dalam menangani *query* sederhana. MySQL dapat memproses lebih banyak SQL per satuan waktu.

e.) Column Types

MySQL memiliki tipe kolom yang sangat kompleks, seperti *signed/unsigned integer, float, double, char, varchar, text, blob, date, time, datetime, timestamp, year, set dan enum.*

f.) Command dan *Functions*

MySQL memiliki operator dan fungsi secara penuh yang mendukung perintah SELECT dan WHERE dalam query.

g.) Security

MySQL memiliki beberapa lapisan *security* seperti level subnetmask, nama host, dan izin akses *user* dengan sistem yang mendetail serta *password* yang menggunakan sistem enkripsi.

h.) Scalability dan *Limits*

MySQL mampu menangani *database* dalam skala besar, dengan jumlah *records* lebih dari 50 juta dan 60 ribu tabel serta 5 milliar baris. Batas index yang dapat ditampung oleh MySQL adalah sebanyak 32 index dari tiap tabel.

i.) Connectivity

MySQL dapat melakukan koneksi dengan *client* melalui penggunaan protokol TCP/IP, Unix soket (Unix), atau Namd Pipes (NT).

j.) Localisation

MySQL dapat mendeteksi pesan kesalahan (*error code)* pada *client* dengan menggunakan lebih dari dua puluh bahasa.

k.) Interface

MySQL memiliki *interface* terhadap berbagai aplikasi dan bahasa pemrograman dengan menggunakan fungsi *Application Programming Interface* (API).

l.) Clients dan *Tools*

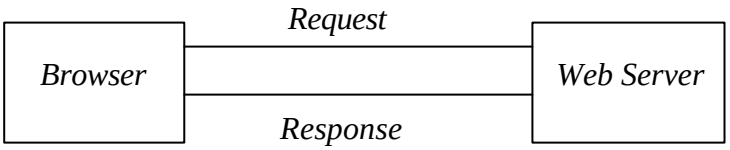
MySQL dilengkapi dengan berbagai tool yang dapat digunakan untuk administrasi *database*. Pada setiap tool dalam MySQL disertakan petunjuk *online*.

m.) Table Stucture

MySQL memiliki struktur tabel yang lebih fleksibel dalam menangani ALTER TABEL, dibandingkan dengan *database* lainnya seperti PostgreSQL atau Oracle.

2.4.3. Arsitektur *Web Database* dengan MySQL

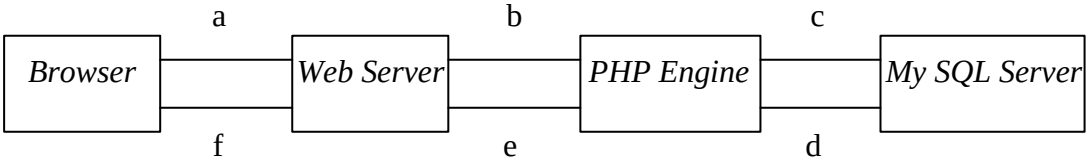
Operasi dasar dari *web server* dapat dilihat pada Gambar 2.4. Sistem ini terdiri dari dua objek yaitu: *web browser* dan *web server*. Kedua objek tersebut mempunyai hubungan yang saling membutuhkan. *Web browser* membuat *request* pada *server* , kemudian *server* mengirim respon kepada *browser.*



Gambar 2. 4.

Hubungan *client/server* antara *web browser* dan *web server*

Aplikasi *web database* yang terdiri dari *web browser, web server, scripting engine,* dan *database server* dapat dilihat pada Gambar 2.5.



Gambar 2.5.

Dasar arsitektur *web database* yang terdiri dari *web browser, web server, scripting engine,* dan *database server*

Penjelasan tahap-tahap dari arsitektur *web database* pada Gambar 2.5. sesuai dengan huruf abjad (a,b,c,d,e,f) adalah sebagai berikut:

a.) *Web browser* melakukan *request* untuk halaman *web* tertentu. Misalnya dengan melakukan *request* untuk *search* suatu *keyword.* Hasil dari *search* tersebut misalnya *result*.php.

b.) *Web server* menerima *request* untuk *result*.php, mengumpulkan file, dan membawa ke PHP *engine* untuk diproses.

c.) PHP *engine* mulai menguraikan *script*. Dalam *script* terdapat perintah untuk menghubungkan ke *database* dan menjalankan *query*. PHP membuat koneksi ke MySQL *server* dan mengirimnya ke *query* yang sesuai.

d.) MySQL *server* menerima *query* dari *database* dan diproses, selanjutnya *result* dikirim kembali ke PHP *engine*.

e.) PHP *engine* selesai menjalankan *script*, yang biasanya mengandung format *query result* pada HTML. *Result* pada HTML dikembalikan ke *web server.*

f.) *Web server* melalui HTML kembali ke *browser,* dimana *user* dapat melihat hasil dari *keyword* yang diinginkan.

2.4.4. Perintah Dasar MySQL

Structured Query Language (SQL) nerupakan bahasa *American National Standard Input* (ANSI) yang digunakan untuk melakukan query data pada *database*. Semua pengoperasian data dapat dikerjakan secara mudah dengan menggunakan bahasa SQL, terutama dalam pemasukan dan seleksi data.

Sebagian besar *software database* mengimplementasikan bahasa ini sebagai komponen utama dari produknya. Contohnya adalah *database server modern* seperti MySQL, PostgreSQL, Oracle, Informix, Sybase, dan lain-lain.

Untuk membuat suatu *database* pada MySQL, maka yang pertama dilakukan adalah membuat *database* baru dengan perintah:

```
Mysql> CREATE DATABASE skripsi;
```

Kemudian akan muncul di layar:

```
Query OK, 1 row affected (0.00 sec)
```

Selanjutnya untuk bekerja pada database tersebut dilakukan perintah:

```
Mysql>USE skripsi;
```

Di layar muncul tulisan:

```
Mysql> Database changed
```

Selanjutnya kita dapat memasukkan perintah-perintah lainnya untuk bekerja pada *database* yang sudah kita buat. Beberapa perintah SQL yang digunakan antara lain:

a.) Membuat tabel baru:

```
Mysql> CREATE TABLE product(
    -> product_code char(10),
    -> category char(20),
    -> product_name char(30),
    -> type char(20),
    -> size char(15)
    -> );
```

Dari perintah di atas maka terbentuk satu tabel dengan nama "product".

b.) Menampilkan data

```
Mysql> SHOW TABLES;
```

Di layar akan muncul:

```
+--------------------+
| Tables in skripsi  |
+--------------------+
| product            |
```

c.) Mengisi data:

```
Mysql> INSERT INTO product
    -> values ('AP6401','Keramik','Granito','Granito Tile', '600x600 mm');
```

Dari perintah di atas, maka dilakukan pemasukan data ke dalam tabel product.

d.) Menampilkan isi tabel

```
Mysql> SELECT product_code, category, product_name, type, size from product;
```

Di layar akan muncul:

```
+------------------+-------------+------------------+---------------+---------------+
| product_code     | category    | product_name     | type          | size          |
+------------------+-------------+------------------+---------------+---------------+
| AP6401           | Keramik     |  Granito         | Granito Tile  | 600x600 mm    |
+------------------+-------------+------------------+---------------+---------------+
```

2.5 *Entity Relationship Diagram* (ERD)

Dalam menggambarkan dan mendesain suatu aplikasi *database* untuk Tugas Akhir ini digunakan *Entity Relationship Diagram.* Semua data-data

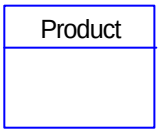
dikumpulkan dan dianalisa sehingga menjadi *conceptual data model*. Pada *conceptual data model* ini diGambarkan ringkasan dari kebutuhan-kebutuhan *user* yang terdiri dari *entity, relationship,* dan *attributes.* Dari *conceptual model* maka dibuat *physical model* . Pada *physical model* ini dilakukan pengGambaran secara lebih detail mengenai tipe dari *atributes* dan *relationship* antara tabel melalui *primary key* dari masing-masing tabel.

2.5.1. Komponen-komponen dasar ERD

Entity Relationship Diagram mempunyai beberapa komponen. Berikut ini merupakan komponen dasar ERD beserta dengan contoh Gambarnya pada Power Designer 6.0. yaitu:

a.) *Entity*

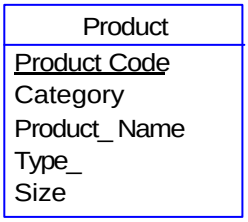
Objek dasar yang direpresentasikan oleh ER diagram adalah *entity. Entity* dapat berupa objek fisik seperti *product*, *customer*, *web admin,* dan lain-lain, atau dapat juga merupakan objek secara konseptual seperti perusahaan, pekerjaan, dan lain-lain, seperti pada Gambar 2.6.



Gambar 2.6. *Entity* ***Product***

b.) Attribute

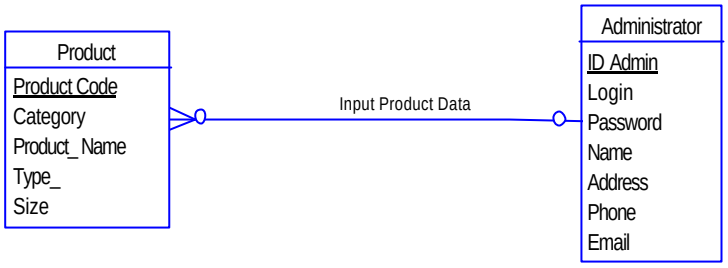
Dalam tiap *entity* terdapat *atribute,* yang merupakan *properties* khusus yang menjelaskan *entity.* Sebagai contoh *entity* produk mempunyai *atributes*: *product_code, category, product_name, type, size.* Salah satu dari *atributes* tersebut dapat menjadi *primary key*. Untuk lebih jelasnya dapat dilihat di Gambar 2.7.



Gambar 2.7. *Atibutes* pada *entity* ***Product***

c.) *Relationship*

Relationship merupakan hubungan yang terjadi antara *entity*, sehingga memudahkan dalam melihat hubungan-hubungan yang ada antara satu *entity* dengan satu atau beberapa *entity* lainnya. Hal ini ditunjukkan oleh Gambar 2.8.



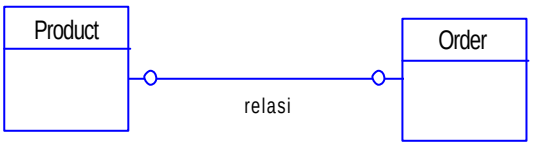
Gambar 2.8.

Relationship antara *entity* ***Product*** dengan *entity* ***Administrator***

Pada *relationship* dikenal dua istilah yaitu *cardinality* dan *ordinality*. *Cardinality* merupakan relasi antara *entity* yang menunjukkan jumlah *instance* pada relasi tersebut. Ada tiga jenis *cardinality* yaitu:

- *One to one*

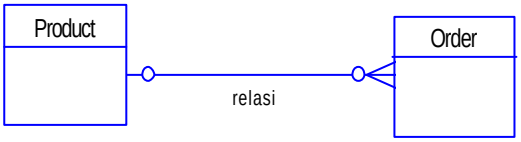
  MengGambarkan relasi satu *instance* di satu *entity* dengan satu *instance* di *entity* yang lain, seperti pada Gambar 2.9.



Gambar 2.9. Relasi *One to one*

- *One to many*

  MengGambarkan relasi antara satu *instance* di satu *entity* dengan beberapa *instance* di *entity* yang lain, seperti pada Gambar 2.10.



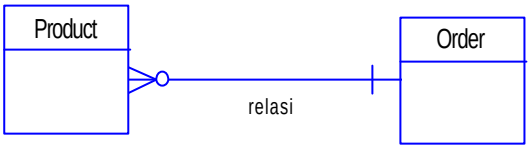
Gambar 2.10. Relasi *One to many*

- *Many to many*

  Menggambarkan relasi antara beberapa *instance* di satu *entity* dengan beberapa *instance* di *entity* yang lain, seperti pada Gambar 2.11.

Gambar 2.11. Relasi *Many to many*

Ordinality menggambarkan status relasi yang dapat berupa wajib/*mandatory* atau pilihan/*optional.* Apabila suatu *entity* harus ada (*mandatory*) dalam hubungan tersebut, maka diberi tanda garis (|), dan apabila hanya berupa pilihan maka diberi tanda bulat kecil (o).



Gambar 2.12. Relasi *Ordinality*

Pada Gambar di atas, *entity "product"* bersifat pilihan pada *entity "order"*. Maksudnya *entity "product"* bisa muncul dan bisa juga tidak muncul pada relasi tersebut, sedangkan *entity "order"* bersifat wajib pada relasi tersebut.

2.6. *Customer Relationship Management* (CRM)

Customer Relationship Management adalah sebuah istilah dalam informasi industri yang digunakan dalam metodologi, software, dan kemampuan internet untuk mengatur relasi dengan *customer* secara terorganisasi. Contohnya, sebuah *enterprise* bisa membangun *database* tentang *customer*nya, yang menjelaskan relasi secara detail sehingga manajemen, *sales, provider service*, bahkan memungkinkan bagi *customer*nya untuk bisa langsung mengakses informasi, membandingkan kebutuhan *customer* dengan rencana produk dan penawaran, mengingatkan *customer* akan *service* yang diinginkan, mengetahui produk apa yang telah dibeli oleh *customer* yang lain, dan sebagainya.

2.6.1. Tujuan *Customer Relationship Management*

Berdasarkan suatu pengamatan industri, maka CRM mempunyai beberapa tujuan antara lain:

a.) Membantu sebuah *enterprise* agar memungkinkan bagi bagian *marketing*nya untuk mengidentifikasi dan mempunyai target yang terbaik bagi *customer*, mengatur proses *marketing* dengan tujuan dan sasaran yang objektif, dan menghasilkan kualitas yang baik untuk *sales team.*

b.) Membantu organisasi untuk mengembangkan *telesales*, *account,* dan *sales management* dengan mengoptimalkan pembagian informasi.

c.) Memungkinkan pembentukan relasi secara langsung dengan *customer* dengan tujuan menambah kepuasaan *customer* dan memaksimalkan pendapatan, mengidentifikasi *customer* yang paling menguntungkan dan memberikan *service* terbaik bagi mereka.

d.) Membekali tenaga kerja dengan informasi dan proses-proses yang penting agar mereka mengetahui *customer*nya, mengerti kebutuhan mereka, dan membangun relasi yang efektif antara perusahaan, *customer,* dan *distributor*.

2.6.2. Memahami CRM melalui Pendekatan Bisnis One-to-one

Saat ini bisnis sedang mengalami revolusi terselubung dalam proporsi yang tak terbayangkan. Dalam suatu bisnis diakui adanya kekuatan individual. Marketing telah berkembang pada level individual dengan bisnis besar maupun kecil.

Untuk saat ini diakui bahwa mempertahankan *customer* yang lama lebih mudah dibanding dengan membawa *customer* baru karena adanya persaingan yang ketat dan pasar yang terlampau besar. Konsekuensinya, filosofi bisnis bergeser dengan sangat cepat dari konsep *"Share of the market"* menuju ke *"share of the customer"*.

Saat ini persaingan yang ada antara satu perusahaan dengan perusahaan lain juga diperhatikan oleh *customer*, oleh sebab itu perusahaan yang menang dalam persaingan adalah perusahaan yang bisa memberikan pelayanan yang baik bagi *customer*nya. *Customer* yang telah merasa puas terhadap produk dan

pelayanan akan memberikan kesempatan pada perusahaan untuk terus menerus memasarkan produknya.

2.6.3. Keuntungan-keuntungan Menerapkan CRM

Ada beberapa keuntungan yang didapatkan oleh perusahaan yang menerpakan CRM dalam bisnis mereka, yaitu:

a.) Memungkinkan perusahaan untuk meningkatkan keuntungan dengan biaya yang minimum.
b.) Menawarkan kesempatan besar kepada perusahaan agar *customer* tetap setia dan percaya kepada perusahaan tersebut walaupun perusahaan tersebut masih termasuk baru.
c.) Memungkinkan kesempatan untuk tetap bertahan di tengah-tengah persaingan bisnis.
d.) Membuat perusahaan mengalami keuntungan yang terus menerus melalui penambahan nilai-nilai kepercayaan *customer* terhadap perusahaan.
e.) Menjaga perusahaan dari resiko *bisnis cycle* (naik-turunnya bisnis).
f.) Mengurangi resiko bisnis yang disebabkan oleh siklus produksi yang singkat.