Modul I

Dasar Pemrograman web

1. Konsep Dasar Internet

Internet terbentuk dari sekumpulan jaringan komputer yang saling tersambung sehingga mampu berinterkasi dan berkomunikasi satu sama lainnya. Untuk dapat berkomunikasi satu dengan lainnya tentu memerlukan suatu bahasa. Bahasa / protokol yang digunakan di jaringan Internet adalah TCP/IP. Dengan bahasa inilah maka tercipta berbagai layanan Internet seperti *e-mail* (*electronic-mail*), http (*hypertext transfer protocol*), ftp (*File Transfer Protocol*) , telnet, irc (*Internet Relay Chat*) dan lain-lain. Tentunya untuk melayani suatu layanan tadi diperlukan sebuah komputer (*server*) yang khusus untuk melayani jenis layanan itu. Seperti layanan *e-mail*, maka diperlukan sebuah komputer yang berperan sebagai *mailserver* untuk melayani e-mail ini. Kemudian untuk layanan http, maka dibutuhkan komputer yang berperan sebagai *webserver* atau ada yang menyebutnya pula *httpserver* dan *httpdaemon* (httpd). Demikian untuk ftp diperlukan *ftp server/fileserver*, irc(*chat*) dibutuhkan *ircsever*.

2. Konsep Dasar *Web Server*

Web Server adalah sebuah atau beberapa komputer yang difungsikan untuk melayani layanan/*service* http (hypertext transfer protocol). Sebuah/beberapa komputer dapat dibuat menjadi *web server* dengan cara menginstall software *web server* seperti APACHE, IIS, PWS, Xitami dan lainnya. APACHE adalah *web server open source* yang paling banyak digunakan di Internet (hampir 60%) , sedangkan IIS (*Internet Information Service*) dari Microsoft digunakan di mesin berbasis NT dan PWS (*Personal Web Server*) digunakan di mesin Win98.

Web Server akan melayani sebuah *request* dari Web Client dengan mengeluarkan *server response* ke *client* berupa transfer *file* HTML yang diterima oleh *browser* di *client*.

3. Pemrograman di Lingkungan *WEB*

Generasi pertama sebuah halaman web adalah di publish secara statis, yang tentu saja mengandalkan HTML, gambar statis, text yang tidak dapat diposisikan secara pas sesuai dengan koordinat yang diinginkan. Tentu saja halaman *web* seperti itu adalah halaman yang sangat *basic* sekali. Untuk mendapatkan hasil yang baik dari teknik tersebut diatas, kita memerlukan tenaga ahli HTML atau seorang designer *web* handal. Kemudian untuk melakukan *update* halaman *web* yang sudah kita buat itu, maka hal yang cukup merepotkan yaitu dengan meng-*edit* HTML itu secara *manual* atau dengan menggunakan *editor*. Selain itu halaman web yang statis tidak kompatibel dengan basis data.

Maka dari itu kita memerlukan halaman *web* yang aktif, tidak statis dan dapat mengakses basis data. Sebuah halaman *web* yang aktif akan selalu berubah-ubah tergantung isi dari basis data ataupun tergantung *event* yang kita buat di level aplikasi. Sehubungan dengan hal itu maka diciptakanlah teknologi pemrograman *web* untuk memungkinkan para pengembanga *web* membuat halaman *web* yang dinamis tersebut. Pada dasarnya kita dapat membagi ke dalam dua bagian pemrograman *web* tersebut, yaitu ***Client Side Programming*** (Pemrograman sisi Client) dan ***Server Side Programming*** (Pemrograman sisi *server*)**.**

3.1 Client Side Programming

Yang dimaksud *client side programming* adalah bahwa eksekusi program yang kita buat itu adalah terjadi di *client*. *Client* disini adalah komputer yang meminta layanan http. Komputer yang meminta layanan http akan menjalankan sebuah *browser* Internet seperti Internet Explorer dari Microsoft ataupun Netscape Navigator dari Netscape. Pemrograman *client side* ini berarti akan mengandalkan *browser* yang dijalankan di *client* karena eksekusi program akan dilakukan di *client* yang berarti di eksekusi oleh *browser* yang dijalankan di komputer *client*. Berikut adalah beberapa teknologi *client side* yang sering digunakan:

- ***ActiveX Controls***
  ActiveX Control adalah program yang berdiri sendiri, yang biasa disebut komponen yang ditulis dalam bahasa C++ atau *Visual Basic*. Ketika ditambahkan ke sebuah halaman *web*, program ini mempunyai beberapa fungsi, seperti *timer*, *client authentication* atau akses terhadap basis data. Obyek *ActiveX control* ditambahkan ke HTML dengan menggunakan *tag* <OBJECT>, yang sekarang ini sudah menjadi standard HTML. *ActiveX control* dapat dieksekusi oleh *browser* atau *server* ketika mereka ditempelkan ke dalam sebuah halaman *web*.
  ActiveX control dikembangkan oleh Microsoft, dan meskipun kompatibel dengan HTML standar, tapi tidak di-*support* oleh Netscape browser yang tanpa menggunakan ActiveX *plug-in*. *ActiveX control* hanya dapat berfungsi di *Internet Explorer* (meskipun begitu beberapa fungsi *ActiveX* disediakan untuk Netscape melalui *plug-in* dari Ncompass).
- ***Java Applets***
  Applets adalah sebuah program yang ditulis dengan menggunakan bahasa pemrograman Java yang bisa dimasukkan ke dalam halaman HTML, sama seperti sebuah *image* dimasukkan ke halaman HTML. Ketika kita menggunakan *browser* dengan Java diaktifkan untuk melihat halaman yang memuat applet, maka kode applet ditransfer ke sistem kita dan dieksekusi oleh browser. Karena applet ditulis dengan Java, maka applet ini mempunyai kelebihan-kelebihan Java, seperti bisa menjadi *stand-alone* dan *cross platform*.
- ***Client Side Script* dan DHTML**
  Client side scripting dikembangkan untuk menyediakan alternatif untuk mengubah HTML yang statis menjadi dinamis. Ketika browser menemukan instruksi-instruksi yang berupa *script* yang ditempel di dalam kode HTML, maka *browser* akan menterjemahkannya ke dalam HTML murni (dengan asumsi *browser* mengerti bahasa *scripting* yang ditemukannya). Ini memungkinkan para pengembang untuk membuat halaman *web* yang interaktif yang lebih banyak mempunyai fungsi daripada sebuah HTML murni.
  JavaScript adalah bahasa *script* yang utama digunakan di Internet. Bahasa *script* ini di-*support* oleh Netscape Navigator (mulai versi 2) dan Microsoft Internet Explorer (mulai versi 3). *Client-Side* VBScript , hanya didukung oleh Internet Explorer, dan maka dari itu bukan merupakan pilihan yang baik dalam scripting Internet sehari-hari, tetapi mungkin kadang-kadang digunakan dalam membuat aplikasi Intranet yang semua produknya berasal dari Microsoft.
  Dynamic HTML adalah *script* yang lebih ditujukan untuk membuat respresentasi dari sebuah halaman HTML. DHTML mempunyai lebih banyak akses ke fitur-fitur seperti kemampuan untuk menganimasikan halaman dan menempatkan gambar grafis dan teks secara tepat dengan menggunakan *absolute positioning.*

3.2 Server Side Programming

Server Side Programming adalah teknik pemrograman *web* dimana kita menulis program di komputer *server* yang kemudian program itu akan dieksekusi di *server* dan hasilnya yang sudah berupa HTML akan dikirim balik ke *client* yang *request* terhadap halaman program yang kita buat tersebut. Beberapa tahun yang silam, satu-satunya solusi agar menampilkan data yang dinamis ke *web* adalah dengan menggunakan sesuatu yang di sebut *Common Gateway Interface* (CGI). Dengan meggunakan program CGI memungkinkan cara yang relatif sederhana untuk dapat menerima *input* dari *user*, melakukan *query* terhadap *database*, dan mengembalikan hasilnya ke *browser*. Baik Microsoft maupun Netscape telah mengembangkan API yang cocok yang bisa digunakan untuk membuat di dalam kode proses untuk melayani *request-request web*. Teknologi *server side* terbaru sekarang ini yang ditawarkan termasuk *Active Server Page* (ASP), *Java*

Servlets, *Java Server Pages* (JSP), *PHP:Hypertext Preprocessor* (PHP), dan masih banyak lagi. Berikut adalah beberapa keunggulan penggunaan *server side scripting* daripada kita hanya menggunakan teknologi *client-side* saja :

- Meminimalisasi *network traffic* dengan membatasi kebutuhan *browser* dan *server* untuk *talk back* diantara keduanya.
- Waktu *loading* akan menjadi lebih singkat karena yang di-*download* hanya halaman yang mengandung HTML saja.
- Masalah kompatibilitas *browser* dapat dihindari.
- Memungkinkan kita memberikan data yang tidak ada pada sisi *client*
- Lebih aman, karena program dieksekusi di *server* sehingga kode program tidak bisa dilihat dari *browser*.

4. Bahasa *Web Scripting PHP:Hypertext Preprocessor* (PHP)

➢ Sejarah PHP

PHP pertama kali dikembangkan pada tahun 1994 oleh Rasmus Lerdorf yang dilepas di Internet. Sedikit demi sedikit orang-orang mulai tertarik dengan apa yang dikerjakan oleh Rasmus ini. Pada tahun berikutnya ia membuat *scripting engine* yang bisa merespon *form* dari *input form* HTML yang merupakan *Form Interpreter* (FI), lalu terciptalah apa yang disebut sebagai PHP/FI atau PHP2.

Beberapa lama kemudian, orang-orang mulai menggunakannya untuk hal-hal yang lebih serius, dan pengembangan pun berubah dari hanya satu orang menjadi satu kelompok para pengembang. Ini merupakan awal dari PHP3. Para grup pengembang ini yang terdiri dari Rasmus Lerdorf, Andi Gustmans, Zeev Suraski, Stig Bakken, Shane Caraveo dan Jim Winsted, memperbaiki dan memperluas *scripting engine*-nya dan menambahkan API sederhana yang memperbolehkan *programmer* lain untuk menambahkan fungsionalitas ke dalam bahasa (PHP) dengan menuliskan sebuah modul untuk itu. Sintaksisnya pun lebih disempurnakan yang dibuat sedemikian rupa sehingga *familiar* bagi orang-orang yang berasal dari pemrograman *object oriented* atau bahasa prosedural. Jika anda pernah belajar C, C++ atau Java, atau pernah belajar shell/awk *scripting*, atau pernah menulis bahasa Pascal atau program Vbasic, maka mempelajari konstruksi dasar PHP akan terasa mudah.

PHP3 pun di-*release* pada bulan juni tahun 1998. Versi 4 dari PHP mulai diterbitkan pada akhir tahun 1999. PHP4 dikembangkan oleh perusahaan resmi yaitu Zend yang menghasilkan beberapa produk pengoptimalisasian PHP4, namun produk ini komersial, sementara *engine core* PHP4 nya sendiri adalah *free*. Saat ini PHP sudah mencapai versi 4.0.5 (di-*release* tanggal 30 April 2001). Dengan segala kelebihannya pengguna PHP terus bertambah setiap bulannya.

➢ Tentang PHP

PHP (merupakan akronim dari : *PHP Hypertext Preprocessor*), adalah bahasa *scripting* sisi server (*server-side*) yang bisa ditempel di HTML (*embedded*). Ini berarti PHP bisa digunakan bersama-sama dengan dokumen HTML sehingga kita bisa membuat HTML itu dinamis sehingga bisa membuat halaman *web* lebih dinamis. Kita bisa membuat aplikasi *web* dengan lebih baik yang tidak sekedar pajangan-pajangan informasi yang susah untuk di-*update*.

Tetapi mengapa memilih PHP?, padahal masih banyak bahasa lain seperti ASP, Cold Fusion, Perl, Java dan lainnya. Penulis memilih PHP sebagai bahasa *server side scripting* karena kesederhanaannya (*simplicity*), PHP mempunyai *native* API untuk koneksi ke berbagai *database* sehingga otomatis koneksinya akan lebih cepat dibandingkan melalui ODBC (*Open Database Conectivity*) yang merupakan perantara antara bahasa *scripting* dengan *database*, sebagaimana yang selalu dilakukan oleh ASP misalnya. Selain itu PHP sangat *cross-platform*, yang berarti dapat berjalan di platform manapun baik di Windows atau pun di Unix. Berikut adalah beberapa keunggulan PHP selengkapnya:

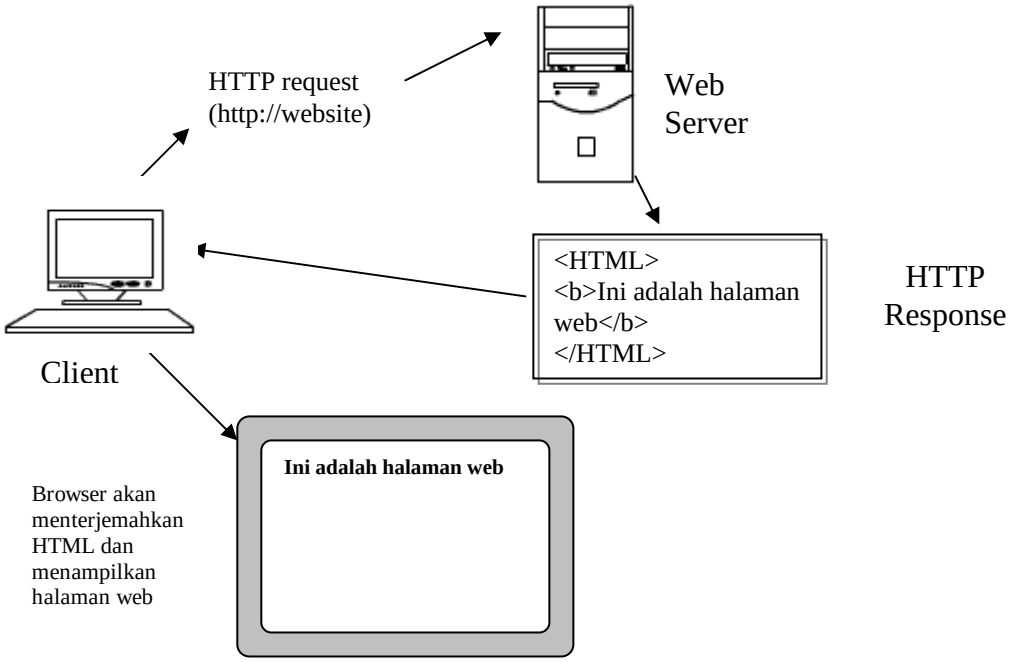
1. Eksekusi scripting dilakukan sangat cepat sehingga meningkatkan *throughput* dari *server*.

2. Simplisitasnya menjadikan penulisan program PHP lebih pendek dan sederhana sekaligus mudah dipahami.
3. Dukungan koneksinya hampir ke semua *database* yang beredar sekarang ini termasuk MySQL, PostgreSQL, mSQL, Sybase, Informix, Interbase, Oracle, SQL Server, Ms Access, dBase dan masih banyak lainnya.
4. Selain menggunakan ODBC, PHP mempunyai *native* API untuk koneksi ke *database*, sehinngga akses ke *database* akan lebih cepat karena mengakses langsung ke *database*-nya (tidak melalui ODBC terlebih dahulu).
5. PHP dapat dipasang di beberapa *web server* seperti PWS, IIS, Apache, Xitami, Netscape Enterprise, AOL server dan Oreilly Wesite Pro. PHP juga dapat dipasang dengan mode CGI atau ISAPI.
6. PHP dapat berjalan di berbagai platform seperti Windows dan Unix. Ini memungkinkan programmer misalnya melakukan *development* di Windows untuk kemudian di pasang di Linux.
7. PHP adalah *software open-source* yang gratis dan bebas didistribusikan kembali di bawah lisensi GPL (*GNU Public License*). Source PHP dan file binernya dapat secara bebas didapatkan di Internet, yaitu di situs resminya : http://www.php.net . Pengembangan PHP yang begitu cepat sehingga bug PHP dapat secara cepat dilaporkan dan diatasi hanya dalam hitungan hari bahkan hitungan jam.

➢ Konsep Pemrograman PHP

Untuk memahami proses atau prosedur pemrograman sisi server PHP maka kita terlebih dahulu harus mengetahui bagaimana sebuah HTML biasa diperlakukan oleh *web server*.Yang terjadi disebuah halaman HTML adalah sebagai berikut: ketika sebuah *request* ke sebuah halaman *web* datang dari browser, maka *web server* melakukan 3 langkah:

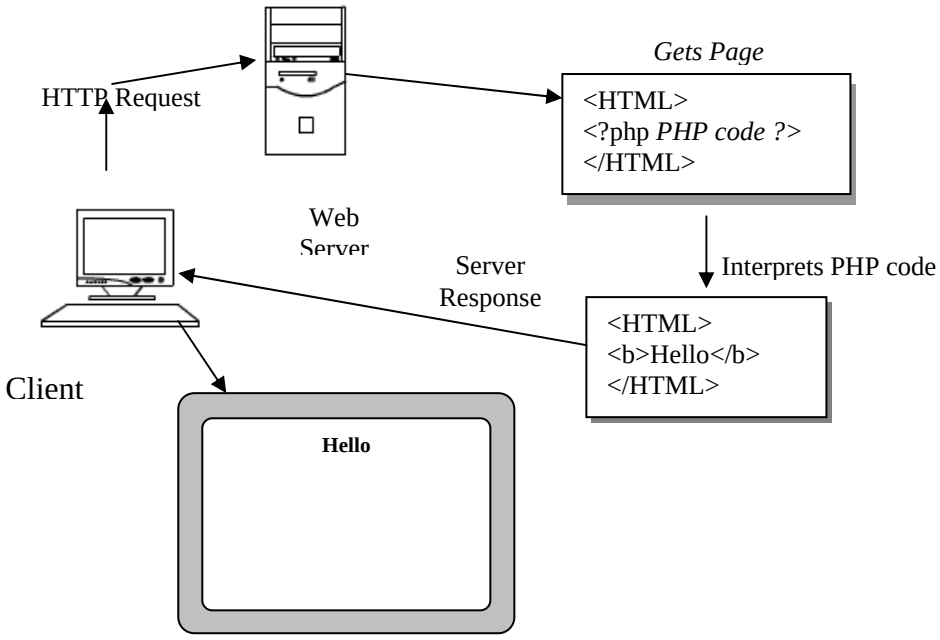
- Membaca *request* dari *browser.*
- Mencari halaman yang diminta di *server.*
- Mengirim balik halaman yang diminta melalui Internet atau Intranet ke *browser*.



Gambar 2.2 Prosedur Web Server memperlakukan *file* html biasa

Sedangkan yang terjadi di sebuah halaman PHP adalah sebagai berikut: Ketika ada *request* ke sebuah halaman PHP maka *server* melakukan hal-hal berikut :

- Membaca *request* dari *browser.*
- Mencari halaman yang diminta di *server.*
- Melakukan instruksi yang ada di halaman PHP yang diminta.
- Mengirim balik halaman hasil eksekusi ke *browser.*



prosedur *web server* memperlakukan *file* PHP

Kode PHP disimpan (*saved*) sebagai sebuah *file plain text* dalam format ASCII, jadi kita bisa menulis kode/program PHP di hampir semua *text editor*, seperti vi, emacs, dan Notepad. Penggunaan *editor* HTML seperti Dreamweaver atau Homesite akan sangat membantu dalam menulis program PHP.

Seperti dijelaskan sebelumnya, program PHP dapat dimasukkan / ditempel di halaman HTML, yang sebelumnya dieksekusi di *server* sebelum dikirim ke *browser*. Contoh berikut ini akan menjelaskannya :

```
<HTML>
<?php
    echo (“Text generated by PHP.”);
?>
</HTML>
```

Sebuah file .html biasa akan dilewatkan begitu saja oleh HTTP *server* (*web server*). *Web server* tidak akan mencoba untuk memproses isinya dan akan langsung dikirim balik ke *browser* karena adalah tugas *browser* untuk memproses file .html seperti itu. File yang berekstensi .php akan diperlakukan berbeda. Pertama *web server* akan mencari kode PHP. Pertama-tama *Web server* akan berjalan dengan “*HTML mode*”. Dengan kata lain pertama kali server menjalankan sebuah file akan mengasumsikan bahwa isi file tersebut hanya berisi HTML, CSS, JavaScript, teks sederhana dan beberapa teks lain yang bisa langsung dilewatkan ke browser tanpa harus diterjemahkan terlebih dahulu. *Web Server* akan masuk ke “PHP mode” ketika *server* menemukan tag php, yang digunakan untuk melakukan ‘*escape*’ atau keluar dari kode HTML. Ada beberapa cara untuk melakukannya :

- Pada contoh diatas, penulis menggunakan instruksi proses XML seperti ini :
  <?php echo (“kode PHP ditulis disini”); ?>
- Juga bisa menggunakan instruksi proses SGML seperti ini:
  <? echo (“kode PHP ditulis disini”); ?>
- Cara yang ini agak sedikit merepotkan, dan pasti sangat familiar bagi para pemrogram JavaScript dan VBScript. Gunakan hanya jika *editor* yang digunakan tidak bisa menangani instruksi pemrosesan:
  <SCRIPT LANGUAGE=’php’> echo (“kode PHP ditulis disini”);</SCRIPT>
- Mulai PHP 3.0.4, kita bisa mengkonfigurasi PHP untuk menggunakan karakter ‘*escape*’ ASP:
  <% echo (“kode PHP ditulis disini”); %>

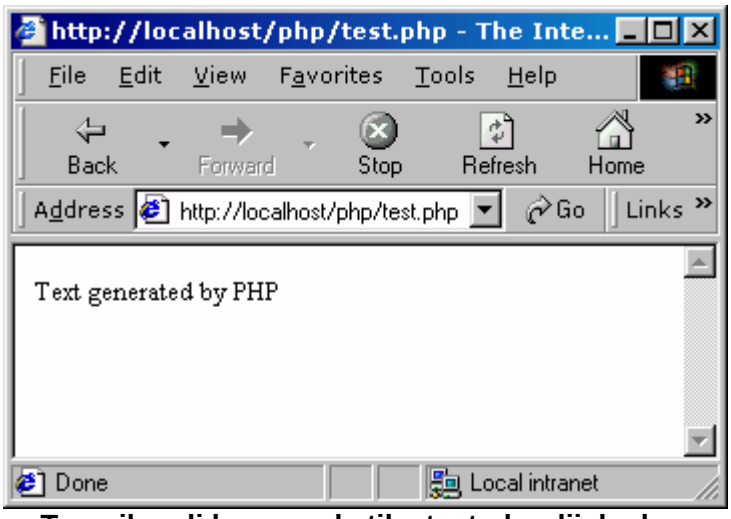
HTTP *server* akan langsung mengerti ketika menemukan karakter ‘*escape*’ diatas. Setelah menemukan mekanisme karakter *escape* seperti diatas, maka *web server* akan langsung memproses kode PHP itu, kemudian hasil dari proses tersebut akan dikirim ke browser berupa HTML biasa. Ketika http server menemukan *closing tag* ?> maka *server* pun akan kembali ke mode HTML dan kembali mengirim dokumen *web* tanpa melakukan suatu proses di sisi server.

Pada contoh diatas *statement* PHP nya adalah :

echo (“Text Generated by PHP.”);

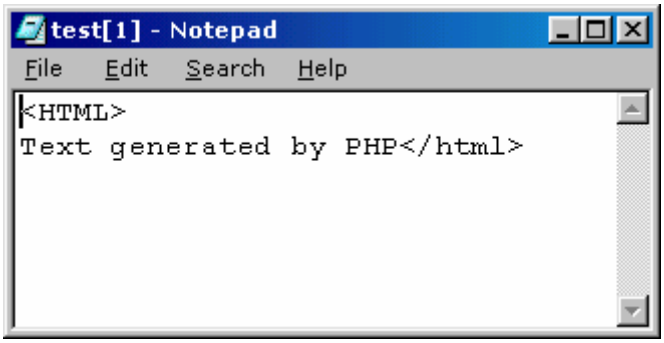
Perlu diperhatikan bahwa setiap diakhir sebuah *statement* PHP harus diakhiri dengan titik koma (;). Jika tidak maka akan muncul pesan kesalahan. *Statement* echo diatas akan menghasilkan *output* ke *browser*. Dalam kasus ini kita mengistruksikan PHP untuk mencetak *output* ke *browser* sebuah string “Text generated by PHP”. (Beberapa programmer PHP menggunakan *statement* print yang fungsinya sama dengan echo).

Statement echo boleh menggunakan tanda kurung ataupun tidak, jadi dua *statement* echo berikut ini akan menghasilkan hal yang sama :
echo (“Text generated by PHP.”);
echo “Text generated by PHP.”;
Simpan contoh diatas dengan nama misalnya ‘test.php’ ke *directory root* atau direktori yang termasuk didaftarkan di *web server*. Jika *file* test.php di akses lewat *browser* maka halaman web ini akan menampilkan tulisan “Text generated by PHP”, seperti yang terlihat dibawah ini :



Tampilan di browser ketika test.php dijalankan

Untuk melihat apa yang terjadi di halaman ini, kita bisa lihat *source code*-nya dengan klik *view->source*:



Gambar 2.5 Tampilan *source code* test.php yang diterima oleh browser

Bisa dilihat bahwa semua perintah PHP beserta tag-nya menghilang, dan hanya terlihat tag yang di luar tag PHP yaitu tag <HTML> dan string yang ingin kita tampilkan. Ini karena *script* PHP dieksekusi di *server*, sebelum halaman ini di kirim ke *browser*.

Variabel di PHP bersifat seperti penampung , jadi tipe data setiap variabel bergantung dari isinya. Setiap variabel ditulis dengan didahului tanda ‘$’. Misalnya :

```
<?
        $username = “Sandi”; // $username bertipe string
        $umur = 21; // $umur bertipe integer
        $umur = 21.5; // sekarang $umur bertipe double
?>
```

5. Instalasi PHP di Windows

Langkah-Langkah Instalasi PHP dan MySQL :

1. Ambil file php-4.0.6-Win32.zip atau download PHP for windows terbaru dari www.php.net.
2. Untuk membuka file .zip *double* klik file tersebut, dengan catatan Anda perlu menginstal Winzip atau program pembuka zip lain terlebih dahulu
3. Ekstrak semua file yang ada didalamnya. Dengan winzip, Anda dapat melakukan sebagai berikut : tekan Extract dan masukan c:\php ke isian Extract To, tekan tombol Extract
4. Copy-kan file php.ini-dist ke direktori Windows Anda (c:\windows atau c:\winnt). Gantilah namanya menjadi php.ini.
5. Buka file php.ini tersebut di Notepad. Carilah string extension _dir dan ubah menjadi :
6. Extension_dir = c:/php/extension
7. Perhatikan tanda pemisah direktori yang dipakai disini yaitu /, bukan \.
8. copy-kan file php4ts.dll didirektori c:\php ke c:\windows\system (untuk Windows 9X/Me) atau c:\winnt\system32 (untuk NT/2000/XP). Lalu bukalah file c:\Program Files\Apache Group\apache\conf\httpd.conf di Notepad dan tambahkan baris berikut dibawah baris LoadModule yang sudah ada :
9. LoadModule php4_module c:/php/sapi/php4apache.dll
10. AddType application/x-httpd-php.php
11. Dan tambahkan baris berikut dibawah baris-baris AddModule yang sudah ada :
12. AddModule modphp4.c
13. Save file dan setelah itu restart Apache , caranya pilih Start Menu > programs > Apache HTTP Server > Control Apache Server > Restart
14. Kini Apache Anda telah memiliki kemampuan skripting PHP. Bagaimana cara mengujinya ? Mari membuat PHP kecil. Buatlah sebuah file dilokasi sbb : s:\Program Files \Apache Group \apache\htdocs\test.php dan isikan :
15. <? echo 1+1; ?>
16. Lalu buka URL http://localhost/test.php di browser. Jika muncul tulisan “2” berarti instalasi berhasil.
17. Sekarang telah memiliki sistem PHP lokal yang bisa Anda pakai untuk belajar PHP, mencoba skrip-skrip PHP yang ada.

Modul II

PHP dan HTML

HTML singkatan dari HyperText Markup Language merupakan bahasa yang digunakan untuk membuat halaman WEB SITE

PHP merupakan sebuah bahasa scripting yang menyatu dengan tag-tag HTML, dieksekusi di server, dan digunakan untuk membuat halaman WEB yang dimamis

STRUKTUR PROGRAM HTML

struktur program HTML adalah sebagai berikut

```
<HTML>
        <HEAD>
                <TITLE> Judul Halaman WEB SITE</TITLE>
        </HEAD>

        <BODY>
        Isi web site dituliskan pada bagian BODY
        </BODY>
```

Simpanlah contoh tersebut dengan nama test.html.
Setiap file HTML harus memiliki paling sedikit bagian seperti diatas, dan disimpan dengan extension HTML atau HTM. Tag HTML biasanya berpasangan, tag pembuka dan penutup menggunakan kata yang sama, bedanya adalah, untuk tag penutup ada tanda slash "/"

Berikut ini contoh-contoh tag HTML

Tag HTML	Kegunaan
<HTML>	Merupakan penanda untuk setiap file HTML
<HEAD>	merupakan bagian Head dari HTML
<TITLE>	Judul dari halaman WEBSITE
<BODY>	Pada bagian ini dituliskan isi dari WEBSITE
<CENTER>	membuat tulisan ditengah halaman
<TABLE>	membuat tabel
<FORM>	membuat form isian

Berikut ini tag HTML yang tidak berpasangan
<P> : paragraph baru

 : ganti baris

Struktur Program PHP

Kode program PHP menyatu dengan tag-tag HTML dalam satu file. Kode PHP Diawali dengan tag <? atau <?PHP dan ditutup dengan tag ?> atau ?PHP> sesuai dengan tag pembukanya.. File PHP berisi tag HTML dan kode PHP diberi ekstensi .PHP. Berdasarkan ekstensi ini, pada saat file diakses, server akan tahu bahwa file mengandung kode PHP. Server akan menerjemahkan kode

ini dan menghasilkan output dalam bentuk tag HTML yang akan dikirimkan ke browser client yang mengakses file tersebut.

berikut contoh program PHP

```
<HTML>
        <HEAD>
                <TITLE> Pemrograman PHP</TITLE>
        </HEAD>

        <BODY>
        <CENTER>
        <?
        echo “Baris ini ditulis dengan kode PHP”;
         ?>
        </CENTER>
        </BODY>
```

simpan contoh diatas dengan nama test.php
Pada file tersebut ada satu baris perintah php yaitu

```
<?
echo “Baris ini ditulis dengan kode PHP”;
?>
```

Perintah dibuka dengan tag <? dan ditutup dengan tag ?> , dan setiap baris perintah PHP diakhiri dengan tanda titik koma (;). Perintah PHP dapat disisipkan disembarang tempat dari file diatas.

Sebuah file PHP dapat juga keseluruhannya berisi perintah PHP. sebagai contoh, file test.php diatas dapat juga dituliskan sebagai berikut:

```
//contoh program php yang menuliskan keseluruhan program php
<?
echo ("
        <HTML>
                <HEAD>
                        <TITLE> Pemrograman PHP</TITLE>
                </HEAD>

                <BODY>
                        <CENTER>
                        Baris ini ditulis dengan kode PHP
                        </CENTER>
                </BODY>
        </HTML>" );
 ?>
```

Spasi tidak berpengaruh pada penulisan baris perintah PHP, hal ini memudahkan dalam pengaturan penulisan program agar program dapat lebih mudah dibaca.

Untuk penulisan komentar program (baris yang tidak dieksekusi oleh program) ada 3 style yang dapat digunakan, yaitu :

1. C style, komentar diawali dengan tag /* dan diakhiri dengan * ?, style ini digunakan untuk komentar yang lebih dari satu baris.

2. C++ style, komentar ini diawali dengan tag // dan hanya berlaku untuk satu baris komentar, untuk baris berikutnya harus diawali dengan tag // lagi
3. Bourne Shell style, diawali dengan tag # untuk satu baris komentar.

Tipe dan Konversi Data PHP

Tipe data dari sebuah variabel akan ditentukan otomatis oleh PHP bergantung pada operasi yang sedang dilakukan menggunakanvariable tersebut. PHP mengenal 5 tipe data yaitu :

1. Integer

   Tipe data integer meliputi semua bilangan bulat, besarnya range integer pada PHP adalah antara
   –147483648 sampai +2147483647 pada platform 32-bit. PHP akan secara otomatis mengkonversi data integer menjadi floating point jika berada diluar range diatas.
   Integer dapat dinyatakan dalam bentuk octal (basis 8), desimal (basis 10) atau heksadesimal (basis 16)

   contoh:

```
$decimal = 16;
$heksadesimal = 0x10;
$octal = 020
$desimal_minus = -16; #bilangan negatif
```

2. Floating Point

   Floating point merepresentasikan bilangan pecahan atau bilangan desimal. Range tipe ini antara 1.7308 sampai 1.7E+308. Floating point dapat dinyatakan dalam bentuk desimal dan dalam bentuk pangkat

   contoh :

```
$decimal = 0.017;
$pangkat= 17.0E-3
```

   Untuk menangani operasi-operasi data bilangan yang membutuhkan tingkat ketelitian tinggi, PHP menyediakan fungsi fungsi BC (Binary Calculation)

3. String

   Sebuah data dengan tipe string dinyatakan dengan mengapitnya menggunakan tanda petik tunggal (‘ ‘) maupun ganda (“ “)
   contoh:

```
$str1 = ‘Ini sebuah string’;
$str2 = “Ini sebuah string”;
```

   Perbedaanya adalah , jika kita menggunakan tanda petik tunggal, maka pada string itu tidak dapat kita masukkan variabel dan escape sequence handling,
   coba jalankan program berikut ini

```
<HTML>
<HEAD>
<TITLE> String dengan tanda petik tunggal dan ganda</TITLE>
</HEAD>
<?
//string01.php
$var=”string”;
$str1=”contoh $var dengan tanda petik ganda”;
$str2=’contoh $var dengan tanda petik tunggal’;
```

```
echo(“<br>$str1”);
echo(“<br>$str2”);
?>
<BODY>
</BODY>
</HTML>
```

Coba jalankan di browser, diperoleh hasil pada string yang diapit tanda petik ganda dapat digunakan variabel $var dan tercetak isinya, yaitu “string”, sebaliknya jika diapit tanda petik tunggal $var tercetak apa adanya
str1 akan menghasilkan : Contoh string dengan tanda petik ganda
str2 akan menghasilkan : Contoh $var dengan tanda petik tunggal.

Escaped Characters

Sequence	**Meaning**
\n	baris baru
\r	carriage return
\t	tab horizontal
\\	backslash
\$	tanda dollar
\”	tanda kutip
\[0-7]{1,3}	Urutan karakterk yang sesuai dengan ekspresi reguler adalah karakter yang berada dalam notasi oktal
\x[0-9A-Fa-f]{1.2}	urutan karakter yang sesuai dengan ekspresi reguler adalah karakter yang berada dalam notasi heksa

4. Arrays
Array adalah sebuah data yang mengandung satu atau lebih data, dan dapat diindeks berdasarkan numeric, maupun string (*associative array*). Data yang dikandung oleh sebuah data array dapat dari tipe data lainnya bahkan juga dapat bertipe array (multiple array). PHP membolehkan elemen dari array terdiri atas data dengan tipe yang berbeda-beda.
contoh:

```
<HTML>
<HEAD>
<TITLE> Contoh tipe data array</TITLE>
</HEAD>
<BODY>

<?
//array01.php
$arr[]=1;
$arr[]=’10’;
$arr[]=”15”;
$arr[]=’A’;
$arr[]=”string”;
$buah[apel] = “merah”;
$buah[jeruk]=”kuning”;
$buah[mangga]=”hijau”;

for ($i=0;$i<5;$i++)
{
echo (“<br> variabel \$arr[$i] nilainya adalah  : $arr[$i] ”);
}
```

```
echo (“<br> variabel \$buah[apel] nilainya adalah  : $buah[apel] ”);
echo (“<br> variabel \$buah[jeruk] nilainya adalah  : $buah[jeruk] ”);
echo (“<br> variabel \$buah[mangga] nilainya adalah : $buah[mangga] ”);



?>

</BODY>
</HTML>
```

Pada contoh diatas, array dideklarasikan dengan langsung memasukkan nilainya. Cara lain adalah dengan menggunakan fungsi array() atau list(). Dengan cara deklarasi langsung, jika angka index (angka dalam kurung) tidak dinyatakan, secara otomatis akan diisi dengan index berikutnya sesuai urutan pengisiannya.

Bila program diatas dijalankan, maka hasil yang didapat adalah sebagai berikut:

variabel $arr[0] nilainya adalah : 1

variabel $arr[1] nilainya adalah : 10

variabel $arr[2] nilainya adalah : 15

variabel $arr[3] nilainya adalah : A

variabel $arr[4] nilainya adalah : string

variabel $buah[apel] nilainya adalah : merah

variabel $buah[jeruk] nilainya adalah : kuning

variabel $buah[mangga] nilainya adalah : hijau

Tipe data string meskipun didefinisikan sebagai salah satu tipe data primitif, tetapi dapat pula dipandang sebagai sebuah array, dan dapat dimanipulasi layaknya sebuah data bertipe array.

contoh

```
<HTML>
<HEAD>
<TITLE> String sebagai array</TITLE>
</HEAD>
<BODY>
<CENTER>

<?
//array02.php
$str=”Data String”;

for ($i=0;$i<11;$i++)
{
echo (“<br> $str[$i] ”);
}

?>
</CENTER>
</BODY>
</HTML>
```

Array dapat juga dideklarasikan dengan mengunakan fungsi array(), contoh berikut menunjukkan bagaimana membuat array 2 dimensi dengan menggunakan fungsi array()

```
$fruits = array(
    "fruits"  =>
array("a"=>"orange","b"=>"banana","c"=>"apple"),
    "numbers" => array(1, 2, 3, 4, 5, 6)
    "holes"   => array("first", 5 => "second", "third")
);
```

5. Objects
   Object adalah sebuah tipe data yang dapat berupa sebuah bilangan, variabel atau bahkan sebuah fungsi. Object dibuat dengan tujuan untuk membantu programmernya yang terbiasa dengan Object Oriented Programming. Meski fasilitas OOP yang disediakan PHP masih sangat kurang.
   Berikut contoh penggunaan tipe data object.

```
<HTML>
<HEAD>
<TITLE> Tipe Data Object</TITLE>
</HEAD>
<BODY>
<?
//object.php
class Test{
    var $str="Variabel Class";
    function set_var($str){
```

```
                $this->str = $str;
        }
}

$class = new Test;
echo $class->str;
$class->set_var("Variabel Object");
echo("<br> $class->str");

?>
</BODY>
</HTML>
```

Dalam program object.php diatas, dibuat sebuah class dengan nama test, kemudian dibuat sebuah data object test dari class test, data ini dicetak dan keluarannya adalah “variabel class”

Class test juga mempunyai sebuah method dengan nama set_var, method ini digunakan untuk memberi nilai pada variabel $str. Data Object test yang dibuat atau instance dari class test akan mempunyai semua properties dari class test, termasuk method-methodnya, sehingga sebuah data bertipe object juga dapat mengandung sebuah method(fungsi). Pada program diatas method set_var pada object test kita pakai untuk mengubah nilai variabel $str menjadi “Variabel Object” kemudian mencetaknya.

Nilai Boolean

Setiap tipe data PHP memiliki nilai Boolean yang spesifik menyertainya, nilai Boolean ini biasanya digunakan pada struktur kontrol program seperti IF atau IF-ELSE

- Untuk integer dan floating point, nilai booleannya adalah false jika nilainya 0, selainnya adalah true
- Untuk string, nilai booleannya adalah false jika string kosong (“”), selainnya adalah true
- Untuk tipe data array, akan bernilai false jika elemennya kosong, dan sebaliknya adalah true
- untuk tipe data object, bernilai false jika tidak ada variabel atau fungsi/method yang terdefinisi didalamnya, dan sebaliknya bernilai true
- untuk variabel yang belum dibuat atau didefinisikan nilainya false.

PHP memiliki 2 keyword true dan false, true merepresentasikan nilai integer 1 dan false merepresentasikan string kosong, Keyword true dan false tidak *case sensitive*.

Konversi Tipe Data

Tipe data ditetapkan sesuai konteksnya apda saat digunakan, tidak ditentukan secara eksplisit

Sebuah variabel $var dideklarasikan bertipe integer, dapat berubah secara otomatis menjadi bertipe floating point jika diberi nilai pecahan.
Contoh:

```
<?
//konversi.php
$var = 1 ;
echo “\$var = “.$var.” dan bertipe ”.gettype($var)  ;
```

```
$var=1 +1.0;
echo(“<br>”);
echo “\$var = “.$var.” dan bertipe ”.gettype($var)  ;

?>
```

Hasilnya adalah sebagai berikut:

$var = 1 dan bertipe integer
$var = 2 dan bertipe double

Perubahan pada tipe data diatas juga dapat dilakukan secara eksplisit yang biasa disebut dengan type casting. Casting pada sebuah variabel dilakukan dengan menuliskan jenis tipe data yang diinginkan di depan nilai dari variabel tersebut

Tabel PHP cast Operators

Operators	Function
(int), (integer)	cast to an integer
(real), (double), (float)	cast to a floating point number
(string)	cast to a string
(array)	cast to an array
(object)	cast to an object

Variabel

- Variable dinyatakan dengan tanda $ dibelakang nama variabel.
- Nama variabel dapat terdiri atas angka, huruf dan underscore.
- Penamaan variabel bersifat case sensitive artinya penggunaan huruf kecil dan huruf besar dibedakan.
- Deklarasi sebuah variabel selalu diikuti dengan pemberian (assignment) nilai dari variabel tersebut, sehingga jika sebuah variabel belum memiliki nilai, tidak perlu dideklarasikan terlebih dahulu.
- Pemberian nilai variabel yg mengacu pada nilai dari variabel lain yang menjadi referensinya dinyatakan dengan tanda & didepan nama variabel referencenya.

Contoh:

```
<?
//var_reference.php
$nama01= “Anita Rachman”;
$nama02 = &$nama01;
echo (“<br> Variabel \$nama01 = $nama01”);
echo (“<br> Variabel \$nama02 = $nama02”);

//nilai variabel $nama01 diubah
echo (“<br> Variabel \$nama01 diubah menjadi \”Budiman\” “ );
$nama01=”Budiman”;
echo (“<br> Variabel \$nama01 = $nama01”);
echo (“<br> Variabel \$nama02 = $nama02”);
?>
```

Hasilnya adalah sebagai berikut:

Variabel $nama01 = Anita Rachman
Variabel $nama02 = Anita Rachman

Variabel $nama01 diubah menjadi “Budiman”
Variabel $nama01 = Budiman
Variabel $nama02 = Budiman

Variabel Scope

Variabel scope bergantung pada konteks dimana variabel tersebut dinyatakan. Variabel yang dinyatakan didalam suatu fungsi akan memiliki scope lokal hanya didalam fungsi itu saja. Variabel yang dinyatakan di dalam bagian sebuah script program akan memiliki scope global dalam script tersebut, variabel dalam sebuah file yang dimasukkan akan juga memiliki sifat global. Contoh:

```
<?
//a.inc
$a = aisyah;
?>

<?
//scope.php

include “a.inc”; //memanggil file lain yang berisikan variabel
$b = “Budi”;

//deklarasikan fungsi
function c() {
   $c = “Cantika”;
   echo (“<br> variabel ini tercetak pada function c”);
   echo(“<br>$a<br>$b<br>$c”);
}
c(); //memanggil fungsi c
echo (“<br> variabel ini tercetak pada badan program”);
echo(“<br>$a<br>$b<br>$c”);
?>
```

Bila program diatas dijalankan, hasilnya adalah sebagai berikut:

variabel ini terletak pada function c

Cantika
variabel ini terletak pada badan program
Aisyah
Budi

Terlihat variabel $a yang terletak pada file a.inc dapat dikenali pada badan program scope.php melalui fungsi include, tetapi tidak dikenali pada function c(). Demikian pula halnya variabel $b yang dideklarasikan pada badan program scope.php tidak dapat dikenali pada function c(). Sebaliknya variabel $c yang dideklarasikan pada function c() hanya dikenali di dalam function c() saja.

Variabel $a dan $b dapat dibuat dikenali dalam function c() dengan memberinya sifat global.

```
<?
```

```
//scope.php

include “a.inc”; //memanggil file lain yang berisikan variabel
$b = “Budi”;

//deklarasikan fungsi
function c() {
   global $a,$b;
   $c = “Cantika”;
   echo (“<br> variabel ini tercetak pada function c”);
   echo(“<br>$a<br>$b<br>$c”);
}
c(); //memanggil fungsi c
echo (“<br> variabel ini tercetak pada badan program”);
echo(“<br>$a<br>$b<br>$c”);
?>
```

hasilnya sebagai berikut:

variabel ini terletak pada function c
Aisyah
Budi
Cantika
variabel ini terletak pada badan program
Aisyah
Budi

Predefined Variabel

PHP memiliki variabel-variabel yang telah didefinisikan sebelumnya, variabel ini nilainya bergantung pada beberapa faktor, seperti jenis server, setting server dan lain-lain. Predefined variabel selengkapnya dapat dilihat dengan memakai fungsi phpinfo()

```
//info.php
<HTML>
<HEAD>
<TITLE> PHP INFO </TITLE>
</HEAD>

<BODY >
<?
phpinfo();
 ?>
</BODY>
</HTML>
```

Variabel dari Form HTML

Form HTML adalah sebuah bagian dari halaman HTML yang memuat elemen kontrol (text, cexbox, radio buttons, menu, dll). Form HTML digunakan untuk mengirimkan data ke server, data ini merupakan variabel input bagi pemrogram PHP yang akan dijalankan. Berikut ini contoh form HTMLdan program PHP yang menangani data yang dikirimkan

```
<?
//form.php
if ($submit) {
  Echo "<BR> Nama  : $nama";
  Echo "<BR> Email : $email";
  Echo "<BR> Jenis Kelamin : $sex";
}
?>

<HTML>
<HEAD>
<TITLE> Contoh Form </TITLE>
</HEAD>
<BODY>

<? Echo "<FORM ACTION=$PHP_SELF METHOD=POST >"; ?>

<P>
Nama : <INPUT TYPE="text" NAME="nama"> <BR>
Email :<INPUT TYPE="text" NAME="email"> <BR>
Jenis Kelamin : <INPUT TYPE="radio" NAME="sex" value= "Pria">
Pria
        <INPUT TYPE="radio" NAME="sex" value= "Wanita">
Wanita <BR>

<INPUT TYPE="submit" NAME="submit" value="Kirim">
<INPUT TYPE="reset">
</FORM>
</BODY>
</HTML>
```

coba simpan file diatas dengan nama form.php, kemudian jalankan di browser.
Pada saat tombol KIRIM di klik, form akan mengirimkan ke file tujuan yang ditunjukkan oleh atribut action. Dalam program diatas, file yang dituju adalah $PHP_SELF yaitu file form.php itu sendiri. Ada 4 variabel yang dikirimkan yaitu, $nama, $email, $sex dan $submit, nama variabel itu sesuai dengan nama elemen kontrol . Pada baris ketiga program form.php terdapat sebuah if kontrol yang memeriksa apakah ada variabel yang dikirimkan. Jika ada, variabel tersebut akan dicetak.

Konstanta

- Konstanta mirip dengan sebuah variabel, hanya saja nilainya tetap dan tidak perlu memakai tag $.
- Konstanta dideklarasikan menggunakan fungsi define().
- Konstanta dideklarasikan hanya satu kali dan nilai tidak dapat diubah atau didefinisikan lagi pada keseluruhan program.

Contoh

```
<?
//konstanta.php
define (“phi”, 3.14);
$r_lingkaran = 5;
$luas_lingkaran = phi * $r_lingkaran * $r_lingkaran;
echo “<br> Jari-jari sebuah lingkaran =$r_lingkaran cm”;
echo “<br> Luas Lingkaran = $luas_lingkaran cm persegi”;
?>
```

Bila program diatas dijalankan, maka hasilnya adalah
Jari-jari sebuah lingkaran = 5 cm
Luas Lingkaran = 78.5 cm persegi

Predefined Konstanta

PHP menyediakan beberapa built-in konstanta, diantaranya
FILE : menyatakan nama file yang sedang diproses
LINE : menyatakan baris program yang sedang diproses
PHP_VERSION : versi PHP yang digunakan
PHP_OS : Sistem Operasi yang digunakan
TRUE : NILai Boolean true
FALSE : Nilai Boolean False

Modul III

Operator, Statement dan Fungsi

Operator

- Operator digunakan untuk melakukan sebuah operasi pada satu atau lebih nilai
- Nilai-nilai pada satu operasi disebut operan
- Pada operasi penjumlahan 1 + 2, nilai 1 dan 2 disebut operan, sedangkan + merupakan operatornya

Operator Aritmatika

Operator aritmatika digunakan untuk operasi-operasi dasar tambah, kurang, bagi.
Berikut ini tabel operator aritmatika

Contoh	Operasi	Hasil
$a + $b	Penjumlahan	$a ditambah $b
$a - $b	Pengurangan	Selisih $a dengan $b
$a * $b	Perkalian	$a dikali $b
$a / $b	Pembagian	$a dibagi $b
$a % $b	Modulus	Sisa hasil $a dibagi $b

Operator Assignment

Assignment adalah operasi pemberian/pendefinisian nilai.
Contoh operator assignment antara lain =, +=, -=, *=, .=

```
<?
//contoh contoh operator assignment
$a = 1 ; //a diberi nilai 1
$a+=1; //ekivalen dengan $a=$a+1 , hasilnya $a bernilai 2
$a -=1; //ekivalen dengan $a=$a-1 , hasilnya $a bernilai 1
$a *=2; //ekivalen dengan $a=$a*2 , hasilnya $a bernilai 2
$a /=2; //ekivalen dengan $a=$a/2 , hasilnya $a bernilai 1
?>
```

Operator Bitwise

Operator bitwise digunakan untuk operasi-operasi bilangan binari
Tabel operator bitwise

Operator	Operasi	Contoh
$a & $b	And	11 (1011 binari) & 13 (1101 binari) ---------- hasilnya 9 (1001 binari)
$a | $b	Or	11 (1011 binari) | 13 (1101 binari) ---------- hasilnya 15 (1111 binari)
$a ^ $b	Xor	11 (1011 binari) ^ 13 (1101 binari) ---------- hasilnya

		6 (0110 binari)
~$a	Not	~ 11 equals 12 (desimal)
$a << $b	Shift left	11 (1011 binari) <<2 hasilnya 2 (10 binari)
$a >> $b	Shift right	11(1011 binari) >>2 hasilnya 44(101100 binari)

Operator Perbandingan

Operator perbandingan digunakan pada struktur control program seperti if, elseif, di mana dilakukan perbandingan antara dua nilai.

Tabel Operator Perbandingan

Contoh	**Operasi**	**Contoh**
$a == $b	Sama dengan	True jika $a sama dengan $b
$a === $b	Identik	True jika $a sama dengan $b, dan memiliki tipe yang sama (PHP4)
$a != $b	Tidak sama dengan	True jika $a tidak sama dengan $b
$a < $b	Lebih kecil dari	True jika $a lebih kecil dari $b
$a <= $b	Lebih kecil atau sama dengan	True jika $a lebih kecil atau sama dengan $b
$a > $b	Lebih besar dari	True jika $a lebih besar dari $b
$a >= $b	Lebih besar atau sama dengan	True jika $a lebih besar atau sama dengan $b

Operator Logika

Tabel operator logika

Contoh	**Operasi**	**Contoh**
$a and $b	And	True jika $a dan $b keduanya bernilai true
$a or $b	Or	True jika salah satu atau keduanya $a, $b bernilai true
$xor $b	Xor	True jika salah satu $a atau $b bernilai true, tetapi false jika keduanya bernilai true
!$a	Not	True jika $a bernilai tidak true

Operator Increment.Decrement

Tabel Operator increment/decrement

Contoh	**Operasi**	**Contoh**
++$a	Pre-increment	Nilai $a ditambah satu, kemudian operasi dilaksanakan
$a++	Post-increment	Operasi dilaksanakan, kemudian $a ditambah satu
--$a	Pre-decrement	Nilai $a dikurangi satu, kemudian operasi dilaksanakan
$a--	Post-	Operasi dilaksanakan, kemudian $a

	decrement	dikurangi satu

Contoh program:

```
<?
//operator increment/decrement
//operator.php
$var = 1;

echo (“<br>(operasi ++\$var) nilai \$var =”.++$var);
echo (“<br>(operasi \$var++) nilai \$var =”.$var++);
echo (“<br>(operasi --\$var) nilai \$var =”.--$var);
echo (“<br>(operasi \$var--) nilai \$var =”.$var--);
?>
```

Hasilnya sebagai berikut:

(operasi ++$var) nilai $var = 2
(operasi $var++) nilai $var = 2
(operasi --$var) nilai $var = 2
(operasi $var--) nilai $var = 2

Operator Presedence

Presedence adalah urutan atau level dari operator. Operator dengan level lebih tinggi akan dieksekusi lebih dulu, contoh $hasil = 8 – 3 * 2
Variabel $hasil akan bernilai 2, bukannya 10, sebab operasi * memiliki level yang lebih tinggi dari operator -, sehingga 3*2 akan dieksekusi lebih dahulu, kemudian 8 dikurangi dengan hasil 3*2

Beberapa operator memiliki level presedence yang sama, untuk kasus seperti ini digunakan aturan associativity dari operasi tersebut, contoh: $hasil = 8 / 4 / 2
Operator bagi (/) memiliki asosiacitivity left (dari kiri ke kanan) sehingga urutan eksekusinya adalah 8 dibagi 4 asilnya sama dengan 2, kemudian dibagi 2 hasilnya sama dengan 1.

Tabel operator precedence

Assocoativity	Operators
Left	,
Left	Or
Left	Xor
Left	And
Right	Print
Left	= += -= *= /= .= %= &= |= ^= ~= <<= >>=
Left	?:
Left	||
Left	&&
Left	|
Left	^
Left	&
Non-associative	== != ===
Non-associative	< <= > >=
Left	<< >>
Left	+ - .
Left	* / %
Right	! ~ ++ -- (int) (double) (string) (array) (object) @

Right	[
Non-associative	New

Statement
Statement berfungsi sebagai rangka dari badan program. Aliran program diatur dengan statemen-statemen struktur kontrol. PHP mengenal dua jenis statemen kontrol, yaitu statemen kondisional dan loop

Statement Kondisional

Statemen kondisional mengatur aliran program berdasarkan kondisi pada tertentu yang ditetapkan. Untuk masalah dengan satu atau dua pencabangan digunakan statemen if dan else, untuk multiple alternatif dapat digunakan elseif dan switch

Statemen If

If digunakan jika satu atau lebih operasi akan dilaksanakan jika syaratnya terpenuhi. Bentuk pernyataannya:

```
<?
if (persyaratan) {
    operasi program;
}
?>
```

Operasi program dilaksanakan jika persyaratan terpenuhi atau bernilai true, jika tidak operasi program akan diabaikan, contoh:

```
<?
//contoh kontrol program menggunakan statemen if
//if.php
$a = 10;
$b = 1
$c = 15
if ($a > $b) { //kondisi if pertama
   echo "$a lebih besar dari $b";
}
if ($a > $c){ //kondisi if kedua
    echo "<BR>$a lebih besar dari $c";
}
?>
```

Hasil dari program diatas adalah :
10 lebih besar dari 1

Kondisi if pertama tepenuhi, 10 > 1 maka kode program dalam kondisi if di proses dan mencetak ke layah, sedangkan kondisi if kedua tidak terpenuhi karena 10 tidak lebih besar dari 15, sehingga baris program dalam kondisi kedua tidak di eksekusi

Kondisional if, dapat dilakukan secara bersarang dengan statemen IF lainnya, hal ini membuat mengeksekusi kondisional statemen lebih fleksibel untuk berbagai kasus dari program.

Statemen IF..ELSE

Pada statemen IF, jika persyaratan tidak terpenuhi atau bernilai false, operasi program akan diabaikan. Namun adakalanya suatu permasalahan memiliki 2 alternatif, di mana jika persyaratan

dipenuhi dilakukan operasi 1, jika tidak dilakukan operasi 2. Untuk masalah seperti ini dapat ditambahkan statemen else pada statemen if, bentuknya sebagai berikut :

```
<?
if (persyaratan){
        Operasi 1;
} else {
         Operasi 2;
}
?>
```

Berikut ini contoh progamnya

```
<?
//contoh kontrol program menggunakan statemen if..else
//ifelse.php
$a = 10;
$b = 1

if ($a > $b) { //kondisi if pertama
   echo "$a lebih besar dari $b";
}else{
   echo "$b lebih besar dari $a";
}
?>
```

Multiple Alternatives

Pada contoh diatas telah ditunjukkan penyelesaian masalah dengan satu atau dua alternatif. Bagaimana dengan masalah dengan lebih dari 2 alternatif atau multiple alternatif. Contoh pada penentuan nilai ujian,
jika nilai ujian >= 85 gradenya A
nilai antara 70 – 84 gradenya B
nilai antara 60 – 69 gradenya C
nilai antara 40 – 59 gradenya D
dan nilai < 40 grade E

Untuk masalah seperti ini dapat digunakan 2 statemen eleseif . Contoh :

```
<?
//masalah penentuan nilai ujian
//elseif.php
$nilai = 83;
if ($nilai >= 85){
   $grade = “A”;
}elseif ($nilai >= 70){
   $grade = “B”;
}elseif ($nilai >= 60){
   $grade = “C”;
}elseif ($nilai >= 40){
   $grade = “D”;
}else{
   $grade = “E”;
}
echo (“<br> Nilai $nilai termasuk grade $grade”);
?>
```

Hasilnya sebagai berikut :

Nilai 83 termasuk grade B

Switch

Switch digunakan pada saat ditemui masalah membandingkan suatu variabel dengan berbagagi nilai, SWITCH statemen mirip dengan serangkaian statemen IF dengan ekspresi yang sama. Pada banyak kesempatan kita perlu untuk membandingkan variable yg sama (atau ekspresi) dengan banyak nilai yg berbeda dan mengeksekusi kode program yg berbeda tergantung pada nilai mana yang memenuhi. Berikut ini contoh 2 cara untuk menuliskan hal yg sama, satu menggunakan statemen IF, dan satunya menggunakan SWITCH statemen

```
<?
//statemen IF
$i = 1;
echo “<br> berikut ini nilai \$i di proses dengan Statemen IF : ”
if ($i == 0) {
    echo "i equals 0";
}
if ($i == 1) {
    echo "i equals 1";
}

if ($i == 2) {
    echo "i equals 2";
}

//statemen SWITCH yang sama dengan statemen IF diatas
echo “<br> berikut ini nilai \$i di proses dengan SWITCH : ”
switch ($i) {
    case 0:
        echo "i equals 0";
        break;
    case 1:
        echo "i equals 1";
        break;
    case 2:
        echo "i equals 2";
        break;
}
?>
```

Hasilnya nilai i = 1 adalah :
berikut ini nilai $i di proses dengan Statemen IF : i equals 1
berikut ini nilai $i di proses dengan SWITCH: i equals 1

Statemen SWITCH dieksekusi baris demi baris (statemen demi statemen). Hanya jika sebuah CASE statemen ditemukan dengan nilai yang cocok dengan nilai dari ekspresi SWITCH, PHP mulai mengeksekusi statemen yang ada. PHP terus mengeksekusi statemen sampai akhir dari SWITCH blok, atau jika menemukan statemen BREAK untuk pertamakalinya. Jika tidak ada statemen BREAK pada akhir daftar CASE , PHP akan terus mengeksekusi statemen pada CASE berikutnya.

Kasus khusus menggunakan CASE default. Nilai default ini memenuhi kondisi yang tidak sesuai dengan CASE yang ada. Contoh CASE default sebagai berikut :

```
<?
//casedefault.php
$i = 10;
switch ($i) {
    case 0:
echo "i equals 0";
    break;
    case 1:
echo "i equals 1";
    break;
    case 2:
echo "i equals 2";
    break;
    default:
echo "i is not equal to 0, 1 or 2";
    }
?>
```

Hasil dari program diatas jika $1= 10 adalah :

i is not equal to 0, 1, 2 ;

Loop

Loop adalah sebuah proses eksekusi program secara berulang-ulang sampai ditemukan kondisi untuk menghakhiri tersebut. PHP mempunyai dua macam loop yaitu while loop dan for loop.

While Loop

While adalah statemen loop yang paling sederhana, bentuknya adalah :

```
while (persyaratan) {
        operasi program;
}
```

Operasi program akan dieksekusi terus menerus selama persyaratan bernilai true.
Contoh :

```
<?
//while.php
$i = 1;
while ($i <= 10) {
    print $i++ ;
}
?>
```

Hasilnya di layar akan menampilkan angka mulai dari 1 sampai 10

12345678910

Do..While

Loop menggunakan do..while pada prinsipnya sama dengan while, yang berbeda adalah pada eksekusi operasi program. Jika menggunakan while persyaratan diperiksa terlebih dahulu. Kemudian jika persyaratan bernilai true, ekseskusi dilakukan. Sebaliknya pada do..while, operasi

program dieksekusi terlebih dahulu baru kemudian persyaratan diperiksa. Jika true, loop diteruskan, jika false, loop dihentikan. Dengan demikian, operasi program minimal dieksekusi sekali.

Contoh :

```
<?
//dowhile.php
 $i = 0;
  do {
      echo “Nilai \$i adalah : $i”;
 } while ($i>0);
 ?>
```

Pada program diatas, menghasilkan : Nilai $i adalah : 0
Nilai $i= 0 tidak memenuhi kondisi $i > 0, akan tetapi karena pemeriksaan do..while diakhir statemen, maka proses cetak sudah dilakukan sebelum diadakan pemeriksaan, selanjutnya, ketika di cek i tidak memenuhi kondisi, program keluar dari loop.

For Loop

Bentuk loop menggunakan statemen for sebagai berikut:

```
for (syarat1,syarat2,syarat3) {
  operasi program;
}
```

Syarat pertama diperiksa sekali saja saat awal mulainya loop. Syarat kedua diperiksa setiap awal iterasi/perulangan. Jika syarat kedua dipenuhi operasi program dieksekusi, jika tidak, loop dihentikan. Syarat ketiga dieksekusi setiap akhir iterasi.

Contoh penggunaan for :

```
<?
//forloop.php

/* example 1 */

for ($i = 1; $i <= 10; $i++) {
    echo "$i";
}
echo "<br>";

/* example 2 */

for ($i = 1;;$i++) {
    if ($i > 10) {
        break;
    }
    echo "$i";
}
echo "<br>";

/* example 3 */

$i = 1;
for (;;) {
```

```
    if ($i > 10) {
        break;
    }
    echo "$i";
    $i++;
}
echo "<br>";

/* example 4 */

for ($i = 1; $i <= 10; print $i, $i++) ;

?>
```

Bila program diatas dijalankan, hasilnya adalah sebagai berikut :
12345678910
12345678910
12345678910
12345678910

ForEach

Foreach digunakan melakukan iterasi pada sebuah nilai array, sintaksnya sebagai berikut:

```
foreach(array_expression as $value){
    operasi program;
}
```

Contoh :

```
<?
//contoh penggunaan foreach
//foreach.php
$a = array(1,3,5,7);
foreach ($a as $v) {
   echo ("<br> $v");
}
?>
```

Hasilnya adalah :

1
3
5
7

Elemen dari array otomatis akan dibaca dari elemen yang pertama, sehingga tidak perlu dilakukan reset.

Break

Break digunakan untuk menghentikan iterasi dari sebuah loop. Break diikuti dengan numeric argumen menunjukkan berapa tingkatan loop yang dihentikan iterasinya.
Contoh :

```
<?
//contoh penggunaan break
//break.php
```

```
$i = 0;
while ($i < 10)
{    echo"$i";
if ($arr[$i] == "stop") {
          break; /*while loop distop*/
}
$i++;}

//menggunakan numeric argumen
$i=0;
while (++$i){

   echo "<br>\$i = $i";
   switch ($i){
    case 5:
      echo"<br> Break Pertama";
      break 1; //switch distop
    case 10:
      echo"<br> Break Kedua";
      break 2; //while distop
    default :
      break;
   }
}
?>
```

Hasil dari program diatas adalah sebagai berikut:
0123456789
$i = 1
$i = 2
$i = 3
$i = 4
$i = 5
Break Pertama
$i = 6
$i = 7
$i = 8
$i = 9
$i = 10
Break Kedua

Break pertama adalah break dari loop switch, sedangkan break kedua adalah break untuk loop while

Continue

Continue digunakan untuk kembali ke awal loop dan sisa operasi program di bawahnya akan diabaikan. Continue diikuti numeric argumen menunjukkan di mana loop akan dimulai kembali.

Contoh:

```
<?
//contoh penggunaan continue
//continue .php
for ($i = 0; $i <5;$i++){
     echo (“ <br> loop ke satu \$i = $i”);
```

```
    for ($j = 0; $j<1;$j++){
        echo (“ <br> loop ke dua \$i = $i”);
       for ($k = 0; $k<1;$k++){
            if ($i==0 || $i==2||$i==4){
                continue 3;
             }
              echo (“ <br> Bagian ini hanya tampil jika \$i bernilai ganjil”);
       }
    }
}
?>
```

Hasilnya :

loop ke satu $i = 0
loop ke dua $i = 0
loop ke satu $i = 1
loop ke dua $i = 1
Bagian ini hanya tampil jika $i bernilai ganjil
loop ke satu $i = 2
loop ke dua $i = 2
loop ke satu $i = 3
loop ke dua $i = 3
Bagian ini hanya tampil jika $i bernilai ganjil
loop ke satu $i = 4
loop ke dua $i = 4

Include dan Require

Include dan require keduanya digunakan untuk memanggil dan mengeksekusi file yang ditentukan. Dengan kedua statemen ini dapat dibuat fungsi-fungsi, konstanta ataupun perintah operasi biasa dalam sebuah file secara terpisah yang dapat dipanggil dari file program lain.

Contoh :

```
<?
//contoh include file berisi konstanta
//pi.inc
$pi =  3.14;
?>

<?
//dolar.inc
$dollar =  9000;
?>

<?
//contoh penggunaan include & require
//ir.php
include"pi.inc";
echo ("<br> pi =  $pi");

require "dollar.inc";
echo ("<br> 1 dolar =  Rp $dollar");
?>
```

Hasilnya adalah :

pi = 3.14
1 dolar = Rp 9000

Include dan require juga dapat digunakan untuk pemanggilan dalam loop.

```
<?
//contoh penggunaan include & require dalam loop
//irloop.php

$arr[] = "pi.inc";
$arr[] = "dollar.inc";

for ($i=0;$i<2;$i++) {
   require $arr[$i];
}
echo("<br> contoh require dalam loop");
echo ("<br> pi = $pi");
echo ("<br> 1 dollar = Rp $dollar");

$arr2[] = "pi.inc";
$arr2[] = "dollar.inc";

for ($i=0;$i<2;$i++) {
   include $arr2[$i];
}
echo("<br> contoh include dalam loop");
echo ("<br> pi = $pi");
echo ("<br> 1 dollar = Rp $dollar");

?>
```

Hasilnya adalah sebagai berikut :

contoh require dalam loop
pi = 3.14
1 dollar = Rp 9000
contoh include dalam loop
pi = 3.14
1 dollar = Rp 9000

Fungsi

- Fungsi adalah sekumpulan perintah operasi program yang dapat menerima argumen input dan dapat memberikan hasil output yang dapat berupa sebuah nilai ataupun sebuah hasil operasi.
- Fungsi dideklarasikan dengan statemen function diikuti nama fungsi dan beberapa variabel input jika ada.
- Nama fungsi tidak boleh sama dengan nama build-in function yang telah dipunyai oleh PHP, jika sama maka akan keluar pesan kesalahan seperti berikut:

Fatal error : Can't redeclare already declared function in *filename* on line … (letak kesalahan)

Contoh deklarasi fungsi:

```
<?
//contoh deklarasi fungsi
//fungsi01.php
$a = 4;
```

```
$b = 5;
$c = $a*kali_dua($b);
echo “Fungsi dengan output nilai”;
echo “<BR>\$c = $c”;
echo “<p>Fungsi dengan hasil operasi”;
cetak_perkalian($a,$b);

//fungsi dengan output nilai
function kali_dua($x){
    return 2*$x;
}

//fungsi dengan output hasil operasi
function cetak_perkalian($x,$y){
    $z= $x*2*$y;
    echo(”<br>\$z = $z”);
}
?>
```

Hasilnya adalah sebagai berikut

Fungsi dengan output nilai
$c=40

Fungsi dengan output hasil operasi
$c=40

Mengirimkan Argumen

- Secara default, argumen dikirimkan ke fungsi adalah berdasarkan nilai, artinya sebuah variabel yang dipakai sebagai argumen input hanya diambil nilainya saja.
- Jika nilai tersebut diubah di dalam operasi fungsi tersebut, nilai asli pada variabel tidak berubah.
- Jika diinginkan sebuah fungsi dapat mengubah nilai variabel digunakan sebagai argumen input, dapat dilakukan dengan mengirimkan argumen ke fungsi sebagai reference.
- Pengiriman argumen berdasarkan reference dilakukan dengan memberi tanda & di depan argumen fungsi tersebut

Contoh mengirimkan argumen :

```
<?
//contoh mengirimkan argumen fungsi berdasarkan reference
//argumen.php

function tambah_kata($tambah) {
    $tambah .=” ini tambahan kata”;
}

$str = “Ini kalimat asli.”;
tambah_kata($str); // argumen dikirim berdasarkan nilai
echo (“<br>$str”);
tambah_kata(&$str); //argumen dikirim berdasarkan referensi
echo (“<br>$str”);
?>
```

Hasilnya adalah sebagai berikut:

Ini kalimat asli.
Ini kalimat asli. ini tambahan kata

Pada pemanggilan fungsi yang pertama, argumen dikirim berdasarkan nilai, sehingga perubahan yang dilakukan didalam fungsi tidak mengubah nilai variabel $str. Pada pemanggilan fungsi yang kedua, argumen dikirim berdasarkan reference, sehingga perubahan nilai argumen yang dilakukan di dalam fungsi juga mengakibatkan berubahnya nilai variabel $str.

Variabel Fungsi

Variabel fungsi adalah sebuah variabel yang berisi sebuah fungsi, contoh :

```
<?
//contoh variabel fungsi
//variabelfungsi.php
function fungsi1() {
    echo("<br> Ini fungsi kesatu");
}
function fungsi2($str) {
    echo("<br> Ini fungsi kedua");
    echo("<br>$str");
}

$var_fungsi = 'fungsi1' ; //variabel fungsi diassign dengan fungsi1
$var_fungsi(); //cara memanggil variabel fungsi sesuai dengan fungsi
yang di assign
$var_fungsi = 'fungsi2'; //variabel fungsi di assign dengan fungsi2
$var_fungsi("Kalimat ini akan ditampilkan");
?>
```

Hasilnya adalah sebagai berikut :

Ini fungsi kesatu

Ini fungsi kedua
Kalimat ini akan ditampilkan

Modul IV

Operasi Array, Operasi File dan Manipulasi String

Operasi Array

Index

Setiap elemen dari array memiliki nomor indeks. Nomor indeks ini diawali dengan angka 0 (nol). Jika pada deklarasi array nomor indeks tidak dideklarasikan secara eksplisit, otomatis deklarasi elemen array yang pertama akan diberi nomor indeks 0, dan deklarasi elemen berikutnya otomatis akan diberi nomor indeks secara berurutan. Tipe data array memiliki pointer yang menunjukkan pada nomor indeks berapa yang saaat itu sedang aktif. Untuk array yang baru dideklarasikan, posisi pointer ini adalah pada nomor indeks yang pertama (indeks 0). Untuk mengetahui nomor indeks elemen di mana pointer tersebut berada (elemen yang sedang aktif) dapat menggunakan fungsi key() dan untuk mengetahui nilai elemen tersebut menggunakan fungsi curent(). Posisi pointer juga dapat diatur, untuk menaikkan menggunakan fungsi next(), untuk menurunkan menggunakan fungsi prev(), untuk meletakkan pointer pada posisi terakhir menggunakan fungsi end() dan untuk mengembalikan pointer pada posisi pertama menggunakan fungsi reset().

Contoh:

```
<?
//contoh penggunaan fungsi-fungsi array
//arrayindeks.php
$murid = array (“rian”, “seno”, “dika”);
echo (“<br> Elemen dicetak dari indeks ke 0 smpai indeks ke 2”);
for ($i=0; $i<count($murid);$i++ ){
     $elemen_aktif = key($murid);
     $nilainya = current($murid);
    echo(“<br> Saat ini pointer berada pada elemen ke $elemen_aktif dan
              nilainya = $nilainya”);
    next($murid);
}
echo(“<p> Elemen dicetak dari indeks ke 2 sampai indeks ke 0”);
end($murid);
for ($i=0; $i<count($murid);$i++ ){
     $elemen_aktif = key($murid);
     $nilainya = current($murid);
    echo(“<br> Saat ini pointer berada pada elemen ke $elemen_aktif dan
              nilainya = $nilainya”);
    prev($murid);
}
end($murid);
$elemen_aktif = key($murid);
echo(“<p> Pointer dipindahkan ke elemen terakhir, yaitu elemen ke
$elemen_aktif”);


reset($murid);
$elemen_aktif = key ($murid);
echo(“<p> Pointer dipindahkan ke elemen pertama, yaitu elemen ke
$elemen_aktif”);
?>
```

Hasilnya:

Elemen dicetak dari indeks ke 0 smpai indeks ke 2
Saat ini pointer berada pada elemen ke 0 dan nilainya = rian
Saat ini pointer berada pada elemen ke 1 dan nilainya = seno
Saat ini pointer berada pada elemen ke 2 dan nilainya = dika

Elemen dicetak dari indeks ke 2 sampai indeks ke 0
Saat ini pointer berada pada elemen ke 2 dan nilainya = dika
Saat ini pointer berada pada elemen ke 1 dan nilainya = seno
Saat ini pointer berada pada elemen ke 0 dan nilainya = rian

Pointer dipindahkan ke elemen terakhir, yaitu elemen ke 2

Pointer dipindahkan ke elemen pertama, yaitu elemen ke 0

Pada loop yang pertama, digunakan statemen for, variabel $i dimulai dari 0 sebab indeks dari array dimulai dari 0, loop dilakukan dengan batas syarat jumlah elemen dari array, untuk memperoleh nilai ini digunakan fungsi count().
Elemen dicetak dari elemen yang pertama sampai yang terakhir, untuk posisi awal tidak ada masalah, pointer sudah ada pada posisi awal, untuk memindahkan pointer ke elemen berikutnya digunakan fungsi next(), sehingga pointer pindah dari elemen ke 0 menjadi posisi elemen ke 1, demikian seterusnya sampai semua elemen array dicetak loop akan berhenti.
Pada loop kedua elemen akan dicetak kebalikan dari loop pertama, yaitu dari indeks terakhir sampai indeks pertama. Sebelum loop dimulai posisi pointer harus diatur pada elemen terakhir menggunakan fungsi end(). Untuk memindahkan pointer ke posisi elemen sebelumnya digunakan fungsi prev().

Pada contoh diatas, indeks elemen array berupa integer. Selain integer indeks array juga dapat berupa string. Untuk merunut array dengan indeks berupa string menggunakan fungsi each(). Fungsi each() mengambil nilai dari array kemudian memajukan pointer satu elemen. Hasil dari fungsi each terdiri atas empat elemen, yaitu 0, 1, key dan val. 0 dan key menunjukkan indeks elemen, 1 dan val menunjukkan nilai elemen. Untuk memasukkan hasil dari fungsi each() ke dalam variabel digunakan fungsi list(). Variabel yang dihasilkan juga bertipe array.
Contoh:

```
<?
//contoh penggunaan fungsi each() dan list()
//arrayeachlist.php

$murid= array(“PIII1” => “rian”, “PIII2” => “seno” , “PIII3” => “dika”);
for ($i=0; $i<count($murid);$i++ ){
     list($key,$val)=each($murid);
     echo (“<br> Saat ini pointer berada pada elemen $key dan nilainya =
$val”);
}
?>
```

Hasilnya :

Saat ini pointer berada pada elemen PIII1 dan nilainya = rian
Saat ini pointer berada pada elemen PIII2 dan nilainya = seno
Saat ini pointer berada pada elemen PIII3 dan nilainya = dika

Variabel array pada contoh arrayindeks.php diubah indeksnya berupa nomor induk murid (berupa string). Paduan fungsi each() dan list() digunakan untuk merunut elemen array. Perhatikan

perbedaannya dengan fungsi key() dam current(), menggunakan fungsi each() pointer secara otomatis maju satu langkah sehingga tidak perlu digunakan fungsi next() lagi.

Urutan Elemen

Operasi urutan(*sorting*) sangat penting dalam operasi array. Salah satu teknik sorting yang sangat populer adalah *bubblesort*. Berikut ini contoh implementasi algoritma bubblesort menggunakan PHP:

```
<?
//contoh program sorting menggunakan algoritma bubblesort
//arraybubblesort.php

$a = array (1,4,2,5,3,9,10,6,8,7);
$b = bubblesort($a, $a2);

echo (“<br>Variabel \$a sebelum diurutkan = ”);
for ($i=0; $i<count($a);$i++ ){
      echo (“$a[$i]”);
}

echo (“<br>Variabel \$a setelah diurutkan = ”);
for ($i=0; $i<count($a);$i++ ){
      echo (“$b[$i]”);
}

function bubblesort($a1,$a2){
   for ($i=sizeof($a1); $i>=1;$i-- ){
         for ($j=1; $j<=$i;$j++ ){
            if($a1[$j-1]>$a1[$j]) {
               $t=$a1[$j-1];
               $t2=$a2[$j-1];
               $a1[$j-1] = $a1[$j];
               $a2[$j-1] = $a2[$j];
               $a1[$j] =$t;
               $a2[$j] =$t2;
            }
         }
   }
return $a1;
}
?>
```

Hasilnya :
Variabel $a sebelum diurutkan = 14253910687
Variabel $a setelah diurutkan = 123456789

Masalah pengurutan elemen array seperti diatas, dapat juga diselesaikan menggunakan fungsi sort(), fungsi sort akan mengurutkan elemen array dari nilai terendah ke nilai tertinggi untuk elemen numerik dan dari a – z untuk elemen string. Sehingga contoh diatas cukup diselesaikan seperti ini:

```
<?
//contoh program sorting menggunakan fungsi sort()
//arraysort.php
$a = array (1,4,2,5,3,9,10,6,8,7);
```

```
echo (“<br> Variabel \$a sebelum diurutkan = ”);
for ($i=0; $i<count($a);$i++ ){
   echo “$a[$i]” ;
}
sort($a);
echo (“<br> Variabel \$a setelah diurutkan = ”);
for ($i=0; $i<count($a);$i++ ){
   echo “$a[$i]” ;
}
?>
```

Fungsi sort() hanya bisa dipakai untuk array dengan indeks numerik. Jika digunakan untuk array dengan indeks string, fungsi sort akan mengubah indeksnya menjadi indeks numerik. Untuk mengurutkan array dengan indeks string digunakan fungsi asort().`

Mengambil dan Menggabungkan Elemen

Fungsi array_slice() digunakan untuk mengambil potongan elemen dari suatu array yang ditunjukkan oleh paramater offset dan parameter length. Cara deklarasi fungsi array_slice() adalah:

Array_slice(variabel, parameter offset, parameter length)

Parameter offset, jika bernilai positif menunjukkan elemen awal pemotongan. Sedangkan jika diberi tanda negatif menunjukkan awal pemotongan dimulai dari elemen akhir dengan jarak sebesar nilai parameter offset.

Parameter length, jika bernilai positif menunjukkan jumlah elemen yang diambil. Sedangkan jika diberi tanda negatif menunjukkan pemotongan diakhiri sejumlah elemen dari elemen akhir yang sebesar nilai parameter length. Jika parameter length tidak dicantumkan, berarti pemotongan dilakukan sampai elemen akhir.

Contoh :

```
<?
//contoh fungsi array_slice()
//arrayslice.php
$a = array (0,1,2,3,4,5,6,7,8,9);
$potongan_pertama = array_slice($a,2);
$potongan_kedua = array_slice($a,2,4);
$potongan_ketiga = array_slice($a,2,-2);
$potongan_keempat= array_slice($a,-4,4);
$potongan_kelima = array_slice($a,-4,2);
echo ("<br> Variabel \$a = ");
for ($i=0; $i<count($a);$i++ ){
   echo "$a[$i]," ;
}
echo ("<br> array_slice($a,2)= ");
for ($i=0; $i<count($potongan_pertama);$i++ ){
   echo "$potongan_pertama[$i]," ;
}
echo ("<br> array_slice($a,2,4)= ");
for ($i=0; $i<count($potongan_kedua);$i++ ){
   echo "$potongan_kedua[$i]," ;
}
echo ("<br> array_slice($a,2,-2)= ");
for ($i=0; $i<count($potongan_ketiga);$i++ ){
   echo "$potongan_ketiga[$i]," ;
}
echo ("<br> array_slice($a,-4,4)= ");
```

```
for ($i=0; $i<count($potongan_keempat);$i++ ){
    echo "$potongan_keempat[$i]," ;
}
echo ("<br> array_slice($a,-4,-2)= ");
for ($i=0; $i<count($potongan_kelima);$i++ ){
    echo "$potongan_kelima[$i]," ;
}

?>
```

Hasilnya :
Variabel $a = 0,1,2,3,4,5,6,7,8,9,
array_slice(Array,2)= 2,3,4,5,6,7,8,9,
array_slice(Array,2,4)= 2,3,4,5,
array_slice(Array,2,-2)= 2,3,4,5,6,7,
array_slice(Array,-4,4)= 6,7,8,9,
array_slice(Array,-4,-2)= 6,7,

Fungsi array_walk

Fungsi array_walk() digunakan untuk melakukan suatu operasi pada setiap elemen bukan array.
Contoh :

```
<?
//contoh fungsi array_walk()
//arraywalk.php
$a = array (0,1,2,3,4,5,6,7,8,9);
echo("<br>\$a = ");
for ($i=0; $i<count($a);$i++ ){
    echo "$a[$i]," ;
}
echo("<p>Setiap elemen variabel \$a dikalikan 5 ");
echo("<br>\$a = ");
array_walk($a,'kali_lima');


function kali_lima($bil){
    $bil1=$bil*5;
    echo "$bil1, ";
}
?>
```

Hasilnya:
$a = 0,1,2,3,4,5,6,7,8,9,

Setiap elemen variabel $a di kalikan 5
$a == 0,5,10,15,20,25,30,35,40,45,

Operasi File

PHP cukup banyak menyediakan fungsi-fungsi build-in untuk operasi file yang sebagian akan dibahas pada bab ini.

Mengakses File

Untuk mengakses sebuah file dari sistem file, HTTP atau FTP digunakan fungsi fopen(). Sintaksnya adalah:

Fopen(nama file, mode akses)

Mode akses menunjukkan operasi yang akan dilakukan pada file tersebut,mode akses antara lain:

- ‘r’ - Membuka file untuk dibaca, pointer diletakkan pada awal file.
- ‘r+’ - Membuka file untuk dibaca dan diubah, pointer terletak pada awal file.
- ‘w’ - Membuka file untuk diubah, pointer diletakkan pada awal file
- ‘w+’ - Membuka file untuk diubah dan dibaca, pointer terletak di awal file.
- ‘a’ - Membuka file untuk diubah, pointer terletak diakhir file
- ‘a+’ - Membuka file untuk diubah dan dibaca, pointer terletak pada akhir file.

Contoh membuka file:

```
<?
//contoh membuka file menggunakan fopen()
//fopen.php
$namafile = "test.txt";
$handlefile=fopen($namafile,r);
echo("<br> $handlefile");
fclose($handlefile);

?>
```

Untuk melihat isi dari file yang telah dibuka digunakan fungsi fget(). Fungsi fgets() akan membaca isi file baris perbaris. Contohnya :

```
<?
//contoh membaca isi file menggunakan fgets()
//fgets.php
$hfile=fopen("fgets.php",r);
while(!feof($hfile)){
          $buffer = fgets($hfile,4096);
         echo("<br> $buffer");
}
fclose($hfile);
?>
```

Hasil dari program diatas adalah sebagai berikut:

```
//contoh membaca isi file menggunakan fgets()
//fgets.php
$hfile=fopen("fgets.php",r);
while(!feof($hfile)){
$buffer = fgets($hfile,4096);
echo("
$buffer");
}
fclose($hfile);
?>
```

Fungsi fgets() membaca isi file baris per baris. Untuk membatasi loop digunakan fungsi feof() untuk memeriksa apakah isi file telah dibaca semua. Setelah membaca isi file, akses ke file tersebut ditutup menggunakan fungsi fclose().
Pada fungsi fgets() jika terdapat tag HTML pada isi file yang dibaca, tag tersebut akan diesekusi, seperti pada contoh program diatas, pada file fgets.php terdapat tag HTML
, sehingga kata berikutnya dicetak pada baris baru.

Untuk mengabaikan tag HTML yang terdapat pada isi file, dapat digunakan fungsi fgetss() untuk membaca isi file tersebut.

Untuk menambahkan isi suatu file dapat digunakan fungsi fputs().

Contoh :

```
<?
//contoh menambahkan isi suatu file menggunakan fputs()
//fputs.php

$handle = fopen("test.txt",a);
fputs($handle,"ini tambahan menggunakan fputs");
fclose($handle);

?>
```

Tambahan isi terletak pada akhir program, sebab pada fungsi fopen() digunakan parameter a sehingga pointer terletak pada akhir file.

Mengkopi File

Untuk mengkopi suatu file digunakan fungsi copy. File yang akan dikopi dapat langsung diambil dari sistem file atau berasal dari form HTML. Contoh program upload file dari client:

```
<HTML>
<BODY>
<?
//contoh mengkopi file menggunakan fungsi copy()
//upload.php
if($submit){
  if (copy($ufile,$ufile_name))
          {echo("<br><center> Copy Sukses");}
  else {echo("<br><center> Copy Gagal");}
}else{
echo ("<FORM METHOD = \"POST\" ACTION = \"$PHP_SELF\"
ENCTYPE=\"multipart/form-data\">");
?>
<CENTER>
<h1> Copy File</h1>
<P><Input type="file" NAME="ufile" SIZE=50>
<P><Input type="submit" NAME="submit" VALUE="COPY">

<?
}
?>
</BODY>
</HTML>
```

Pada contoh program diatas file akan dikopi ke direktory C

Form HTML juga dapat digunakan untuk mengupload lebih dari satu file (multiple file), nama variabel yang dikirimkan dapat dengan nama yang berbeda, atau dengan satu nama sehingga variabel yang dikirim bertipe array.

Untuk mengubah nama sebuah file digunakan fungsi rename() dan untuk menghapus file digunakan fungsi unlink(). Berikut ini contoh mengopi 3 file, dimana file ke dua diubah namanya dan file ketiga dihapus

```
<HTML>
<BODY>
<?
//contoh mengkopi file , menghapus dan mengubah nama file
//kopiubahhapus.php
```

```
if($submit){
  if
(copy($ufile1,$ufile1_name)&&copy($ufile2,$ufile2_name)&&copy($ufile3,
$ufile3_name))
          {echo("<br><center> Copy Sukses");}
  else   {echo("<br><center> Copy GAGAL");}

  if (rename($ufile2_name,"namabaru.php")){
            echo("<br><center> File kedua menjadi namabaru.php");
  }else  {
            echo("<br><center> rename GAGAL ");
  }
  if (  unlink($ufile3_name)){
       echo("<br><center> File ketiga dihapus");
  }else  {
            echo("<br><center> hapus file GAGAL ");
  }

}else{
echo ("<FORM METHOD = \"POST\" ACTION = \"$PHP_SELF\"
          ENCTYPE=\"multipart/form-data\">");
?>
<CENTER>
<h1> Copy File, Ubah File dan Hapus File</h1>
<P><Input type="file" NAME="ufile1" SIZE=50>
<P><Input type="file" NAME="ufile2" SIZE=50>
<P><Input type="file" NAME="ufile3" SIZE=50>

<P><Input type="submit" NAME="submit" VALUE="COPY">

<?
}
?>
</BODY>
</HTML>
```

Download File

Untuk melakukan download file menggunakan fungsi readfile()
Contoh:
File HTML untuk mendownload file:

```
<HTML>
<HEAD>
<TITLE> DOWNLOAD FILE</TITLE>
</HEAD>

<BODY>
<CENTER>
<h1>Download File</h1>
<ahref=”download.php?file=test.txt”>Test.txt</a>
</CENTER>
</BODY>
</HTML>
```

Berikut ini file download.php

```
<?
//download.php
$total = “C:/home/unwim/www/”.$file;
Header(“Content-Type:application/zip”);
Header(“Content-Length:”.filesize($total));
Header(“Content-Disposition: attachment; filename=$file”);
readfile($file);
?>
```

Manipulasi String

Pada bagian ini akan dibahas fungsi-fungsi untuk manipulasi string dan regular expression

Substr()

Fungsi substr() digunakan untuk mengambil potongan dari sebuah string yang panjangnya ditentukan oleh parameter start dan length. Sintaknya adalah sebagai berikut:

substr(string string, int start, int[length])

Parameter start menunjukkan pengambilan dimulai dari karakter ke berapa (karakter pertama dari string adalah karakter ke-0). Jika parameter start diberi tanda minus (-) berarti awal pengambilan ditentukan dari akhir string sebanyak nilai yang ditunjukkan parameter start.
Parameter length menunjukkan banyaknya karakter yang diambil dari variabel string. Jika parameter length diberi tanda minus (-) menunjukkan pengambilan karakter diakhiri sebanyak jumlah karakter yang dinyatakan parameter length, dari karakter terakhir. Jika parameter length tidak dideklarasikan, pengambilan karakter dilakukan sampai karakter terakhir.
Contoh:

```
<?
//substr.php
$str=”abcdefgh”;
$a[]=substr($str,2);
$a[]=substr($str,-4);
$a[]=substr($str,2,4);
$a[]=substr($str,2,-4);
$a[]=substr($str,-4,2);
$a[]=substr($str,-4,-2);

for ($i=0; $i<count($a);$i++ ){
    echo “<br>$a[$i],” ;
}
?>
```

hasilnya:
cdefgh
efgh
cdef
cd
ef
ef

Operasi substr() ini hampir sama dengan opeasi array_slice() pada operasi arry

substr_replace()

Fungsi substr_replace() digunakan untuk menggantikan sebagian atau seluruh karakter dari sebuah string dengan satu atau lebih karakter. Sintaksnya adalah:

substr_replace(string string, string pengganti, int start, int[length])

Parameter start menunjukkan string yang akan digantikan dimulai dari karakter ke berapa (karakter pertama dari string adalah karakter ke 0). Jika parameter start diberi tanda minus (-) berarti awal string yang akan digantikan ditentukan dari akhir string sebanyak nilai yang ditunjukkan parameter start.

Parameter length menunjukkan karakter yang akan digantikan. Jika diberi tanda minus akhir penggantian dihitung dari karakter terakhir.
Contoh :

```
<?
//substr_replace.php
$str=”abcdefgh”;
$tambahan =”ijkl”;
$a[]=substr_replace($str,$tambahan,0);
$a[]=substr_replace($str,$tambahan,0, -(strlen($str)));
$a[]=substr_replace($str,$tambahan,2);
$a[]=substr_replace($str,$tambahan,-4);
$a[]=substr_replace($str,$tambahan,2,4);
$a[]=substr_replace($str,$tambahan,2,-6);
$a[]=substr_replace($str,$tambahan,-4,2);
$a[]=substr_replace($str,$tambahan,-4,-2);

for ($i=0; $i<count($a);$i++ ){
   echo “<br>$a[$i],” ;
}
?>
```

hasilnya :
ijkl
ijklabcdefgh
abijkl
abcdijkl
abijklgh
abijklcdefgh
abcdijklgh
abcdijklgh

Chop(), trim(), ltrim()

Fungsi chop(), trim(), ltrim() digunakan untuk menghilangkan spasi, trim() menghilangkan spasi dikiri dan kanan, ltrim() menghilangkan spasi di sebelah kiri, chop() menghilangkan spasi di sebelah kanan. Selain spasi trim() dan ltrim() juga menghapus whitespace characters lainnya, yaitu : “\n”, “\r”,”\t”, “\v”, “\0”
Contoh:

```
<?
//chop.php
$str="pemrograman   ";
echo chop($str);
echo trim($str);
echo ltrim($str);
echo "<br>";
echo ltrim($str);
echo trim($str);
echo chop($str);
?>
```

Hasilnya:

pemrogramanpemrogramanpemrograman
pemrograman pemrogramanpemrograman

explode(), implode(), split(), join()

Fungsi explode() digunakan untuk memecah sebuah string menjadi beberapa string dalam suatu pola tertentu, sintaksnya:

explode(pola, string)

Fungsi split() relatif sama dengan fungsi explode, hanya saja terdapat optional parameter pembagi untuk menentukan string akan dipecah menjadi berapa bagian, Jika parameter tidak dideklarasikan, pemecahannya akan dilakukan sesuai kondisi string yang akan dipecah, sintaksnya :

split(pola, string, parameter pembagi)

Fungsi implode() dan join() digunakan untuk hal yang sama, yaitu kebalikan dari fungsi explode().

Contoh:
Buat file test.txt yang isinya sebagai berikut:

```
PHP pada dasarnya adalah
sebuah alat untuk membuat
halaman WEB yang dinamis
<br>seperti halnya
Microsoft's Active Server Pages (ASP)
atau JavaServer Pages (JSP)
<p> versi pertama dari PHP
dibuat oleh Resmus Lerdorf
pada tahun 1995
```

Berikut contoh untuk menggunakan fungsi explode(), split() dan implode()

```
<?
//contoh pemecahan string
//explode.php
$fh=fopen("test.txt",r);
$str=fread($fh,1024*40);
$str1=explode(" ",$str);
for ($i=0;$i<count($str1);$i++){
  echo"$i=$str1[$i]: ";
}
echo ("<P> Dicetak sepuluh kata saja");
$str2=split(" ",$str,10);
for ($i=0;$i<count($str2);$i++){
  echo"$i=$str2[$i]: ";
}
echo ("<P> Digabung lagi");
$str3=implode(" ",$str1);
echo"<br> $str3";
?>
```

Hasilnya adalah sebagai berikut :
0=PHP: 1=pada: 2=dasarnya: 3=adalah sebuah: 4=alat: 5=untuk: 6=membuat halaman: 7=WEB: 8=yang: 9=dinamis
seperti: 10=halnya Microsoft's: 11=Active: 12=Server: 13=Pages: 14=(ASP) atau: 15=JavaServer: 16=Pages: 17=(JSP)

: 18=versi: 19=pertama: 20=dari: 21=PHP: 22= dibuat: 23=oleh: 24=Resmus: 25=Lerdorf pada: 26=tahun: 27=1995:

Dicetak sepuluh kata saja0=PHP: 1=pada: 2=dasarnya: 3=adalah sebuah: 4=alat: 5=untuk: 6=membuat halaman: 7=WEB: 8=yang: 9=dinamis
seperti halnya Microsoft's Active Server Pages (ASP) atau JavaServer Pages (JSP)

versi pertama dari PHP dibuat oleh Resmus Lerdorf pada tahun 1995:

Digabung lagi
PHP pada dasarnya adalah sebuah alat untuk membuat halaman WEB yang dinamis seperti halnya Microsoft's Active Server Pages (ASP) atau JavaServer Pages (JSP)

versi pertama dari PHP dibuat oleh Resmus Lerdorf pada tahun 1995

strip_tags(), nl2br()

Kedua fungsi di atas digunakan untuk menangani tag-tag HTML. Fungsi strip-tags() digunakan untuk menghapus tag HTML dari sebuah string. Fungsi nl2br() digunakan untuk mengubah baris baru menjadi tag HTML
.
Contoh :

```
<?
//strip.php
$fh=fopen("test.txt",r);
$str=fread($fh,1024*40);
echo ("*****Tampilan Asli ****<br>$str");
$str=strip_tags($str);
echo ("<P>*****Tag HTML dihilangkan ****<br>$str");
$str=nl2br($str);
echo ("<P>*****Menggunakan  fungsi nl2br()****<br>$str");
?>
```

Hasilnya adalah sebagai berikut:

*****Tampilan Asli ****
PHP pada dasarnya adalah sebuah alat untuk membuat halaman WEB yang dinamis seperti halnya Microsoft's Active Server Pages (ASP) atau JavaServer Pages (JSP)
versi pertama dari PHP dibuat oleh Resmus Lerdorf pada tahun 1995

*****Tag HTML dihilangkan ****
PHP pada dasarnya adalah sebuah alat untuk membuat halaman WEB yang dinamis seperti halnya Microsoft's Active Server Pages (ASP) atau JavaServer Pages (JSP) versi pertama dari PHP dibuat oleh Resmus Lerdorf pada tahun 1995

*****Menggunakan fungsi nl2br()****
PHP pada dasarnya adalah
sebuah alat untuk membuat
halaman WEB yang dinamis
seperti halnya
Microsoft's Active Server Pages (ASP)
atau JavaServer Pages (JSP)
versi pertama dari PHP
dibuat oleh Resmus Lerdorf
pada tahun 1995

Modul V

DATABASE MYSQL

1. Penggunaan *Database Server* untuk mendukung aplikasi *web*

Apabila kita menginginkan sebuah sajian informasi yang fleksibel dan selalu terkini di situs *web* yang kita buat, atau mungkin menyajikan informasi yang pengguna perlukan saja, maka kita akan menyadari bahwa kita memerlukan suatu media penyimpanan tetap untuk menyimpan data yang terstruktur.

Kita bisa menggunakan *file text-delimited* untuk mendapatkan informasi dengan menggunakan *script* untuk mengaksesnya. Untuk kasus yang sederhana, penggunaan *file text-delimited* ini masih memadai, tetapi jika menginginkan sebuah *web* dinamis yang baik tentunya kita lebih cenderung menggunakan sebuah basis data relasional, yang berarti kita harus menggunakan sebuah *SQL database* yaitu sebuah *database engine* yang mengimplementasikan spesifikasi standar *Structured Query Language*. Di dalam basis data relasional, data disimpan di sebuah himpunan tabel-tabel. Setiap tabel berisi satu kolom atau lebih yang mendeskripsikan atribut-atribut data, dan setiap barisnya berisi data-datanya.

Relational Database Management System (RDBMS), telah digunakan selama bertahun-tahun di dunia bisnis dan akademis dan telah terbukti sebagai sistem yang stabil dan dengan metode dan teknik yang solid, memudahkan para pengembang untuk merancang dan membuat aplikasi basis data yang dapat memenuhi kebutuhan.

Selain RDBMS ada juga yang disebut *Object Oriented DBMS* (ODBMS) yang sangat fleksibel dan bersesuaian secara alami dengan struktur data. Dalam *database server* ini data di representasikan dengan sebuah obyek dengan *property* dan *method* yang dapat diaplikasikan. Meskipun ODBMS dapat merepresentasikan data dan relasinya dengan lebih baik, tetapi masih terdapat kekurangan yaitu ODBMS masih bermasalah dalam hal kinerja. Algoritma pencarian di RDBMS sudah matang dan kuat, sementara di ODBMS masih dalam tahap pengembangan. Contoh ODBMS populer adalah termasuk ObjectStore (http://www.odi.com/odilive/), Versant (http://www.versant.com/) dan GemStore (http://www.gemstone.com/).

Selain dua jenis DBMS diatas masih ada lagi DBMS lain yaitu ERDBMS (*Extended Relational Databases*) atau pernah disebut juga ORDBMS (*Object Relational Database*) karena DBMS model ini mempunyai karakteristik RDBMS dan ODBMS. Contoh DBMS jenis ini adalah PostgreSQL (http://www.postgresql.org).

Kebutuhan penggunaan basis data tidak hanya untuk membuat dokumen yang dinamis, tetapi juga penting bagi keinginan orang-orang untuk mendapatkan informasi yang “hidup” setiap harinya melalui antar muka yang sederhana. Kita bisa membuat orang membeli barang dari inventori yang basis datanya itu disimpan di tempat yang berbeda dan didalam komputer yang berbeda sistem operasinya pula dengan sistem *desktop* yang user pakai.

Dengan menggunakan *webserver* yang dikonfigurasi dengan baik, sebuah *database server* (misalnya MySQL), dan dengan beberapa “keajaiban PHP”, satu-satunya hal yang dibutuhkan user adalah sebuah *web browser*. Kita juga bisa mengatur basis data mana saja yang boleh diakses oleh user-user tertentu.

Dengan menggunakan basis data sebagai *backend*, web site yang kita buat akan lebih fleksibel dan kompleksitas dapat dihilangkan. Arsitektur dari sebuah aplikasi basis data berbasis *web* terdiri dari bagian-bagian seperti :

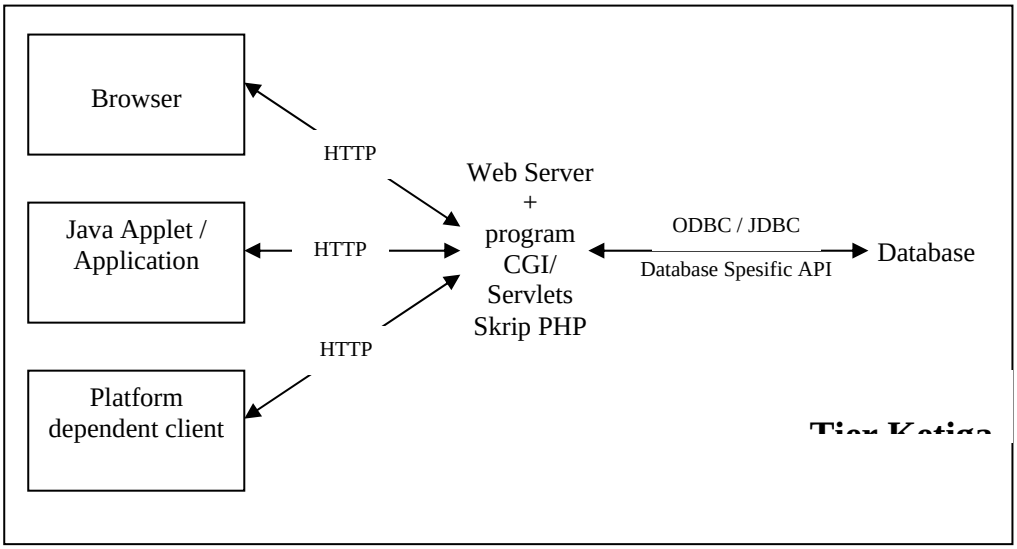
- *Client* : para pengguna web browser, sebuah java applet, aplikasi java atau bahkan platform program client yang dependen.
- *Application Logic*: dikodekan dengan algoritma menggunakan skrip CGI, *special modules web server* atau mungkin sebuah aplikasi server dependen.
- *Database Connectivity* : API (Application Programming Interface) dari *database* atau konektivitas protokol biasa seperti ODBC atau JDBC.
- *Database Server* : RDBMS, ODBMS, dan lain-lain.

Untuk mengimplementasikan sebuah aplikasi bisa dilakukan dengan menggunakan model *multi-tiered* karena satu lapisan atau lebih dapat disatukan nantinya. Implementasi yang biasa dilakukan adalah dengan menggunakan sistem *three-tier:*

1. *Tier* pertama: *web client* (contohnya pengguna *web browser*).
2. *Tier* kedua: web server, skrip CGI dan koneksi ke database dengan menggunakan API (contohnya webserver Apache dengan modul mod_php, yang mendukung database MySQL dan skrip PHP).

3. *Tier* ketiga: server database (contohnya MySQL server).

Tiga tier yang telah disebutkan diatas bisa digambarkan sebagai berikut :



Gambar 2.6. Sistem three-tier

2. Pemilihan MySQL sebagai *database server*

MySQL server adalah *server database* yang kecil/ringan, kompak dan mudah digunakan. Sangat ideal untuk aplikasi kecil dan menengah. MySQL tersedia untuk berbagai *platform* seperti untuk UNIX, Windows NT dan Windows 95/98. Jika di pasang di platform UNIX, MySQL menggunakan *threading* yang memungkinkan kinerjanya yang tinggi dengan skalabilitas yang tinggi pula untuk sebuah *database server*.

MySQL adalah software yang gratis (*free*) di bawah lisensi GNU Public License. Keterangan mengenai lisensi ini dapat dilihat di http://www.gnu.org/. Sedangkan informasi dan software MySQL dapat diperoleh di situs *web* http://www.mysql.com. Beberapa alasan penulis untuk memilih MySQL sebagai *database server* adalah sebagai berikut:

- **Mendukung Standar yang telah ada.**
  MySQL mendukung ANSI SQL92 entry-level dan standar ODBC level 0-2.
- **Dukungan berbagai bahasa.**
  Database server mysqld (*mysql daemon*) dapat memberikan pesan *error* dalam berbagai bahasa seperti bahasa Cekoslowakia, Belanda, Inggris, Estonian, Perancis, Jerman, Hungaria, Italia, Norwegia, Polish, Portugis, Spanyol, dan Swedia. MySQL secara *default* menggunakan set karakter ISO-8859-1 (Latin 1) untuk data dan *sorting.* Set karakter tersebut dapat diubah ketika kita melakukan *compiling*.
- **Mempunyai API untuk berbagai bahasa pemrograman di *client* untuk mengakses *database*.**

Aplikasi-aplikasi basis data MySQL dapat ditulis di berbagai bahasa pemrograman seperti C, Perl, PHP dan lainnya.

- **Mampu membuat Tabel berukuran sangat besar.**

Tabel-tabel di MySQL disimpan di direktori dan file terpisah. Ukuran maksimum dari setiap tabel adalah diantara 4 GB sampai dengan ukuran file yang dapat ditangani oleh sistem operasi yang dipakai.

- **Kecepatan, kehandalan dan kemudahan dalam penggunaannya.**

MySQL lebih cepat tiga sampai empat kali daripada *database server* komersial yang beredar saat ini. MySQL juga dapat dengan mudah di atur (*easy to manage*). Tidak memerlukan seorang *Database Administrator* yang ahli untuk mengatur administrasi pemasangan MySQL.

- **Lebih Murah.**

MySQL adalah RDBMS (basis data relasional) yang *open source*. MySQL didistribusikan dengan gratis tanpa biaya untuk UNIX *platform*, OS/2 dan *Windows platform*. Jadi ongkos untuk menggunakan MySQL akan jauh lebih murah daripada menggunakan database komersial.

- **Melekatnya integrasi PHP dengan MySQL**

Meskipun PHP banyak mendukung beberapa database server lainnya seperti Oracle, SQL Server, PostgreSQL dan lain-lain, tetapi penulis lebih memilih MySQL karena selain database server ini sering digunakan para *programmer* PHP, juga keterikatan yang sangat kuat antara PHP dengan MySQL yang sama-sama software *open-source*, membuat koneksi yang terjadi adalah lebih cepat jika dibandingkan dengan menggunakan *database server* lainnya. Modul MySQL di PHP sekarang sudah di buat *built-in* sehingga tidak memerlukan konfigurasi tambahan pada *file* konfigurasi php.ini.

Meskipun begitu, karena MySQL masih tergolong software yang baru , dan masih dalam tahap perkembangan yang sangat cepat, maka ada beberapa hal yang tidak didukung oleh MySQL server saat ini. Tetapi beberapa fitur dalam *database* yang tidak didukung oleh MySQL ini tidaklah begitu penting untuk kebanyakan aplikasi *web*. Beberapa fitur yang tidak didukung oleh MySQL diantaranya adalah:

- ***Sub-Select***

Sub-Select tidak didukung oleh MySQL. Sebagai contoh, *statement* dibawah ini akan mengembalikan data dari para pekerja (employees) yang gajinya melebihi rata-rata di setiap departemennya:

```
SELECT deptno, ename, sal
    FROM emp x
    WHERE sal > (SELECT AVG(sal)
            FROM emp
            WHERE x.deptno = deptno)
    ORDER BY deptno;
```

Kebanyakan dari statemen sub-select dapat ditulis kembali tanpa menggunakan sub-select. Statemen sub-select yang rumit dapat ditulis ulang dengan memanfaatkan atau membuat *temporary table*.

- ***Stored Procedures dan Triggers***

Sebuah *stored procedure* adalah kumpulan *statement* SQL yang di *compile* dan di simpan di *server*, sehingga *client* dapat menggunakan stored procedure ini, karena statemen SQL-nya sudah di-*compile* dan di-*parsed* di *server* sehingga akan lebih sedikit data yang dikirim dari *client* ke *server*. Sedangkan *Trigger* adalah sebuah *stored procedure* yang akan dipanggil atau dijalankan berdasarkan suatu kejadian atau *event*. Sebagai contoh sebuah *trigger* dapat di set untuk dipanggil ketika dilakukan operasi *UPDATE* terhadap tabel tertentu. *Trigger* juga dapat di set untuk mengirim *e-mail* kepada orang-orang yang tertarik terhadap barang tertentu, dimana barang itu mengalami penurunan harga sebanyak 20% misalnya.

- ***Foreign Key***

Foreign key adalah sebuah kolom pada sebuah tabel yang menunjukkan bahwa kolom tersebut adalah *primary key* pada tabel lain. Ini digunakan untuk memelihara integritas / relasi antar tabel. MySQL tidak mendukung *foreign key* dengan berbagai alasan, diantaranya bahwa dengan adanya penggunaan *foreign key* akan menurunkan kecepatan secra drastis pada operasi INSERT dan UPDATE. Tidak lama lagi MySQL akan dibuat untuk menyimpan definisi *foreign key* pada setiap *database* sehingga *client* dapat mengetahui bagaimana relasi antar tabel yang ada di database dilakukan.

- ***Views***

Views adalah representasi data yang dibuat sedemikian rupa dari satu tabel atau lebih (atau dari view lain). View dapat dianalogikan sebagai *"virtual table"*. *View* tidak memerlukan ruang penyimpanan. Sebagai contoh jika kita mempunyai tabel pekerja (employee) dan menginginkan semua user yang

bukan manajer, hanya dapat melihat nama dan id employee dari tabel pekerja tersebut. Kita bisa membuat view untuk kasus tersebut dengan statemen SQL berikut ini:

Create,=88 View Employee_View as SELECT name, employee-id FROM employee

Dengan membuat hak akses SELECT ke Employee_View ke semua user (selain manajer) maka semua user tersebut hanya dapat melihat nama dan id saja.

MySQL saat ini masih belum mendukung views, tetapi ini merupakan hal yang akan ditambahkan para pengembang MySQL dikemudian hari.

3. Bahasa SQL yang diterapkan di MySQL

Structured Query Language (SQL) adalah bahasa pemrograman standar untuk mengakses dan memanipulasi informasi dari sebuah basis data relasional. SQL merupakan standar ANSI dan ISO, dan di dukung hampir oleh semua basis data relasional.

Pada bagian ini akan dijelaskan statemen-statemen SQL yang digunakan oleh MySQL.

- ***CREATE***

Ini digunakan untuk membuat (*create*) sebuah basis data atau tabel didalam basis data yang telah ada. Sintak untuk membuat sebuah tabel adalah cukup kompleks karena harus mendefinisikan setiap tipe data pada tiap-tiap kolomnya. Untuk membuat sebuah database, sintaks yang dipakai cukup sederhana yaitu:

CREATE DATABASE nama_database

Ada 2 cara lain untuk membuat sebuah database di MySQL, yaitu melalui mysqladmin atau dengan menggunakan PHP. Misalnya sekarang kita akan membuat database dengan nama “mahasiswa”, kita bisa menulisnya:

% mysql

mysql>CREATE DATABASE mahasiswa;

mysql>QUIT;

atau:

% mysqladmin CREATE mahasiswa

Berikut ini adalah variabel sintaks untuk perintah CREATE TABLE:

CREATE [TEMPORARY] TABLE [IF NOT EXIST] tbl_name (create_definition,…) [table_options] [select_statement]

Create_definition:

Col_name type [NOT NULL | NULL] [DEFAULT default_value] [AUTO_INCREMENT] [PRIMARY KEY] [reference_definition]

atau PRIMARY KEY (index_col_name,…)

atau KEY [index_name] (index_col_name,…)

atau INDEX [index_name] (index_col_name,…)

atau UNIQUE [INDEX] [index_name] (index_col_name,…)

atau [CONSTRAINT symbol] FOREIGN KEY index_name (index_col_name,…)
[reference_definition]

atau CHECK (expr)

type:

TINYINT[(length)] [UNSIGNED] [ZEROFILL]
atau SMALLINT[(length)] [UNSIGNED] [ZEROFILL]
atau MEDIUMINT[(length)] [UNSIGNED] [ZEROFILL]
atau INT[(length)] [UNSIGNED] [ZEROFILL]
atau INTEGER[(length)] [UNSIGNED] [ZEROFILL]
atau BIGINT[(length)] [UNSIGNED] [ZEROFILL]
atau REAL[(length,decimals)] [UNSIGNED] [ZEROFILL]
atau DOUBLE[(length,decimals)] [UNSIGNED] [ZEROFILL]
atau FLOAT[(length,decimals)] [UNSIGNED] [ZEROFILL]
atau DECIMAL[(length,decimals)] [UNSIGNED] [ZEROFILL]
atau NUMERIC[(length,decimals)] [UNSIGNED] [ZEROFILL]
atau CHAR(length) [BINARY]
atau VARCHAR(length) [BINARY]
atau DATE
atau TIME
atau TIMESTAMP
atau DATETIME
atau TINYBLOB
atau BLOB
atau MEDIUMBLOB
atau LONGBLOB
atau TINYTEXT
atau TEXT
atau MEDIUMTEXT
atau LONGTEXT
atau ENUM(value1,value2,value3,…)
atau SET(value1,value2,value3,…)

index_col_name:

col_name[(length)]

reference_definition:

REFERENCES tbl_name [(index_col_name,…)]
[MATCH FULL | MATCH PARTIAL]
[ON DELETE reference_option]

[ON UPDATE reference_option]

reference_option:
RESTRICT | CASCADE | SET NULL | NO ACTION | SET DEFAULT
table_option:
TYPE = {ISAM | MYISAM | HEAP}
atau AUTO_INCREMENT = #
atau AVG_ROW_LENGTH = #
atau CHECKSUM = {0 | 1}
atau COMMENT = “string”
atau MAX_ROWS = #
atau MIN_ROWS = #
atau PACK_KEYS = {0 | 1}
atau PASSWORD = “string”
atau DELAY_KEY_WRITE = {0 | 1}

select_statement:
[IGNORE | REPLACE] SELECT … (some legal statement)

Kata kunci TEMPORARY akan membuat tabel sementara / temporer yang secara otomatis akan dihilangkan ketika koneksi ke *database* terhenti. IF NOT EXIST digunakan untuk menghindari kesalahan jika nama tabel yang akan dibuat sudah ada di *database*.

Contoh pembuatan tabel sederhana dengan statemen CREATE:

```
CREATE TABLE buku (
  id          INTEGER UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
  meta        TEXT,
  title       VARCHAR(200),
  authors     VARCHAR(200),
  summary     TEXT,
  keywords    VARCHAR(300),
  body              MEDIUMTEXT,
  published   DATE,
  updated     TIMESTAMP,
  comment     TEXT
);
```

MySQL mempunyai banyak tipe data untuk setiap kolom. Tipe data ini dapat dijelaskan dengan melihat tabel dibawah ini:

Tipe Data	Range Signed	Range Unsigned	Keterangan

TINYINT	-128 sampai 127	0 sampai 255	Tipe integer yang sangat kecil
SMALLINT	-32768 to 32767.	0 to 65535	
MEDIUMINT	-8388608 to 8388607	is 0 to 16777215	
INT / INTEGER	-2147483648 to 2147483647	0 to 4294967295.	
BIGINT	-9223372036854775808 to 9223372036854775807	0 to 18446744073709551615	
FLOAT	-3.402823466E+38 to -1.175494351E-38, 0 dan 1.175494351E-38 to 3.402823466E+38	-	
DOUBLE / REAL	-1.7976931348623157E+308 to -2.2250738585072014E-308, 0 and 2.2250738585072014E-308 to 1.7976931348623157E+308	-	
DATE	'1000-01-01' to '9999-12-31'		Dengan format YYYY-MM-DD
DATETIME	'1000-01-01 00:00:00' to '9999-12-31 23:59:59'		Ditulis dengan format 'YYYY-MM-DD HH:MM:SS'
TIMESTAMP	'1970-01-01 00:00:00' to sometime in the year 2037		
TIME	'-838:59:59' to '838:59:59'		
YEAR[(2|4)]	1901 to 2155, and 0000		
CHAR(M) [BINARY]	M = 1 sampai 255		Tipe karakter dengan panjang tetap sebanyak M
VARCHAR(M) [BINARY]	M = 1 sampai 255		Tipe karakter dengan panjang maksimum M.
TINYBLOB TINYTEXT	Panjang maksimum sebanyak 255 (2^8 – 1)		Tipe BLOB dan TEXT. BLOB bisa berupa string atau biner (menyimpan file gambar misalnya)

BLOB TEXT	Panjang maksimum 65535 (2^16 - 1)		
MEDIUMBLOB MEDIUMTEXT	Panjang maksimum 16777215 (2^24 - 1)		
LONGBLOB LONGTEXT	Dengan panjang maksimum 4294967295 (2^32 – 1)		

Tabel 2.2. Tipe Data MySQL

- ***DROP***

DROP digunakan untuk membuang / menghapus *database*, tabel , index dan function, dengan perintah seperti DROP DATABASE, DROP TABLE, DROP INDEX, dan DROP FUNCTION. Disini hanya akan dibahas DROP DATABASE dan DROP TABLE saja.

Untuk menghapus (*drop*) sebuah database (harap berhati-hati karena ini akan menghapus seluruh elemen dari database itu termasuk tabel beserta data-datanya), bisa menggunakan :

DROP DATABASE [IF EXIST] nama_database

Untuk melakukan *drop* terhadap *database* ini juga dapat dilakukan dengan berbagai cara.

Contoh jika kita ingin men-drop database mahasiswa, kita bisa menggunakan:

% mysql

mysql> drop database mahasiswa;

mysql> quit;

atau

% mysqladmin drop mahasiswa

Untuk melakukan drop terhadap tabel bisa digunakan perintah yang hampir sama:

DROP TABLE [IF EXIST] nama_tabel [, nama_tabel,…]

Selain itu kita bisa melakukan drop terhadap beberapa tabel sekaligus dengan cara :

DROP TABLE mahasiswa, dosen;

- ***INSERT***

Statemen Insert digunakan untuk mengisi atau menambahkan data ke sebuah tabel di sebuah database. Sintaks umumnya adalah sebagai berikut:

INSERT [LOW_PRIORITY | DELAYED] [IGNORE] [INTO] nama_tabel [(nm_kol,…)] VALUES (ekspression,…),(…),…

atau

INSERT [LOW_PRIORITY | DELAYED] [IGNORE] [INTO] nama_tabel [(nm_kol,…)]

SELECT …

atau

INSERT [LOW_PRIORITY | DELAYED] [IGNORE] [INTO] nama_tabel [(nm_kol,…)]

SET nm_kol=expression, nm_kol=expression, …

MySQL dapat melakukan insert dalam dua cara, yang pertama yaitu dengan memasukan kolom secara eksplisit (INSERT … VALUES, atau INSERT … SET) atau dengan mendapatkan *value*-nya dari *database* yang telah ada (INSERT … SELECT).

Opsi LOW_PRIORITY digunakan jika kita menginginkan pemasukan data ketika tidak ada *client* lain yang sedang membaca tabel yang akan dimasukkan.

Jika ada *client* yang tidak akan menunggu sampai proses pemasukan (insertion) selesai, maka kita bisa menggunakan DELAYED. Dengan menggunakan opsi DELAYED maka akan menunda proses pemasukan data sampai tidak ada *client* lain yang membaca tabel yang akan dimasukan datanya, tetapi kontrol program akan segera dikembalikan ke *client*. Keuntungan dengan menggunakan DELAYED adalah bahwa setiap *client* yang mengirimkan *statement* INSERT maka akan dieksekusi sebagai sebuah blok. Ini akan berguna dalam penulisan informasi log misalnya.

Opsi IGNORE digunakan untuk kasus dimana *statement* INSERT berpotensi mengalami kegagalan karena sebuah aturan yang diberlakukan seperti tidak boleh ada nilai ganda di sebuah kolom. Jika terjadi *error* di *database* maka proses INSERT tidak akan dilakukan dan tidak ada data baru yang dimasukkan ke tabel.

Contoh penggunaan INSERT :

INSERT INTO author (id, fullname, email)

VALUES (‘j001’, ‘John Winter’, ‘jw@somesite.com’);

Atau

INSERT INTO author

SET id=’j001’, fullname=’John Winter’, email=’jw@somesite.com’;

Atau jika kita membuat tabel temporer tempdocs yang diisi dengan dokumen yang dibuat oleh John Winter yang diisi sejak awal tahun 1999 dari tabel document:

INSERT INTO tempdocs (id, meta, title, summary, keywords, published)

SELECT id,meta,title,summary,keywords,published FROM document WHERE published >= ‘1999-01-01’;

Untuk menjalankan contoh terakhir ini tentu saja harus dibuat dulu tabel temporer tempdocs dan tabel document.

- ***REPLACE***

Sintaks REPLACE yang hanya ada di MySQL ini hampir sama dengan *statement* INSERT dan penulisannya pun hampir sama. Perbedaannya adalah jika ada data lama yang merupakan UNIQUE INDEX di tabel dan data itu sama dengan data yang akan diisikan ke tabel, maka data lama itu akan di-*replace* dengan data yang baru tersebut.

```
REPLACE [LOW PRIORITY | DELAYED]
        [INTO] nama_tabel [(col_name,...)]
        VALUES (expression,...)
```

Atau

```
REPLACE [LOW PRIORITY | DELAYED]
        [INTO] nama_tabel [(col_name,...)]
        SELECT ...
```

Atau

```
REPLACE [LOW PRIORITY | DELAYED]
        [INTO] nama_tabel
        SET col_name=expression, col_name=expression,...
```

- ***UPDATE***

Untuk merubah data yang telah ada di tabel kita bisa menggunakan statemen update ini. Sintaksnya adalah :

```
UPDATE [LOW_PRIORITY] nama_tabel SET col_name1 = expr1, col_name2 = expr2,...
    [WHERE where_definition] [LIMIT #]
```

Jika kita tidak memakai *statement* WHERE maka operasi *update* akan melakukan *updating* ke semua kolom yang disebutkan. Atau kita bisa menggunakan LIMIT untuk menentukan jumlah perubahan maksimum dari operasi update ini.

Contoh penggunaan UPDATE:

```
UPDATE documents SET title='Table of Contents',
    Comment='Fixed type in the title' WHERE id=231;
```

Statemen diatas akan melakukan pembaruan (*update*) kolom title di tabel document dimana id-nya adalah 231, dan menambahkan *comment* yang menunjukkan alasan diadakannya *update*.

- ***DELETE***

Statement ini akan menghapus satu atau lebih data / *record* yang memenuhi kondisi yang ditentukan. Jika WHERE tidak disertakan maka akan menghapus seluruh data dari tabel, tetapi tentu saja tabelnya tidak ikut terhapus. Jadi penggunaan *statement* ini haruslah dilakukan dengan hati-hati. Sintaksnya adalah :

```
DELETE [LOW_PRIORITY] FROM nama_tabel
```

[WHERE where_condition] [LIMIT rows]

Berikut adalah contoh penggunaan DELETE yang akan menghapus 10 data pertama di tabel document:

DELETE FROM document LIMIT 10;

Jika kita ingin menghapus seluruh artikel yang di terbitkan sebelum 1 Januari 1999 :

DELETE FROM document WHERE published < '1990-01-01'

- ***SELECT***

Statement ini digunakan di aplikasi web ketika akan melakukan pencarian data di database MySQL. Karena itu sintaks ini merupakan sintaks yang rumit:

SELECT [STRAIGHT_JOIN] [SQL_SMALL_RESULT] [SQL_BIG_RESULT]

[HIGH_PRIORITY]

[DISTINCT | DISTINCTROW | ALL]

select_expression,...

[INTO_OUTFILE 'file_name' export_options]

[FROM table_references

[WHERE where_definition]

[GROUP BY col_name,...]

[HAVING where_definition]

[ORDER BY {unsigned_integer | col_name | formula} [ASC | DESC] , ...]

[LIMIT [offset,] rows]

[PROCEDURE procedure_name]]

Penulisan opsi harus berurutan sesuai dengan yang dituliskan diatas, jadi opsi HAVING harus ditulis sebelum opsi ORDER BY atau sebelum opsi LIMIT dan PROCEDURE.

Statement SELECT tidak hanya untuk mendapatkan data dari *database*, tetapi bisa juga untuk mendapatkan hasil dari operasi matematik, seperti :

SELECT SQRT((144 % 5) – 1);

Ini akan mengembalikan nilai 1.732051.

Berikut akan dijelaskan beberapa opsi yang dapat digunakan bersama-sama dengan *statement* SELECT:

DISTINCT

Ini digunakan untuk menjamin bahwa hasil yang didapat tidak terdapat baris yang sama.

INTO OUTFILE

Opsi ini digunakan jika kita menginginkan hasil yang didapat dari query disimpan di sebuah file teks. Secara default ini akan menghasilkan file data dengan field atau kolomnya dipisahkan dengan tab, dengan menggunakan *special character backslash* (\) dan setiap barisnya diakhiri dengan karakter *newline* (\n). Untuk membuat file CSV bisa menggunakan sintaks berikut ini:

... INTO OUTFILE 'outfile.cvs' FIELDS TERMINATED BY ',' ENCLOSED BY '""'

ESCAPED BY '\\' LINES TERMINATED BY '\n' ...

FROM

Digunakan untuk menunjukkan tabel yang sedang digunakan dalam proses SELECT, dan tabel itu dapat dibuat *alias*-nya yang sangat berguna untuk membandingkan field yang namanya sama dari tabel yang berbeda :

```
SELECT field1, field2 FROM mytab AS tab1, mytab AS tab2
    WHERE tab1.id = tab2.subid
```

WHERE

Opsi ini digunakan sebagai pengkondisian dari proses SELECT yang dilakukan, dan dapat berupa perbandingan (*comparison*), fungsi matematika dan ekspresi logika. Contoh berikut akan mendapatkan nama author dan judul bukunya dari tabel *document*, *author* dan *authorsofdoc* yang semua authornya memiliki nama belakang 'SMITH', dan tinggal di kode area 212, dan yang mempunyai artikel yang diterbitkan diantara 1 Januari 1999 dan 15 Juni 1999:

```
SELECT DISTINCT document.title, author.fullname FROM document, author, authorsofdoc
    WHERE author.phone LIKE '212$' AND
        document.published >= '19990101' AND
        document.published <= '19990615' AND
        authorsofdoc.authorid = author.authorid AND
        authorsofdoc.docid = document.docid AND
        author.fullname LIKE "% Smith";
```

LIKE

Merupakan fungsi untuk menyamakan dengan *pattern* atau bentuk yang disebutkan. Contohnya kita bisa menggunakan karakter *wild card* seperti '%' yang berarti akan sama dengan karakter apapun baik ada banyak karakter ataupun tidak ada karakter sama sekali. Dengan menggunakan '_' berarti bentuk ini akan sama dengan satu karakter.

GROUP BY

Opsi ini digunakan untuk mengelompokkan hasil dari query dengan sembarang field. Biasanya berguna ketika kita menggunakan fungsi agregat di dalam parameter seleksi. Sebagai contoh, untuk mendapatkan nilai rata-rata dari umur pekerja bangunan di departemen yang berbeda dari suatu organisasi, kita bisa menggunakan:

```
SELECT AVG(age), deptno FROM employees WHERE job = 'bangunan'
    GROUP BY deptno;
```

HAVING

Pengkondisian pada statemen ini mirip dengan yang digunakan oleh klausa WHERE. Perbedaannya adalah HAVING ini dipakai setelah *statement* GROUP BY. Contoh :

```
SELECT AVG(age) FROM employees WHERE job='bangunan'
    GROUP BY deptno HAVING avg(age) >= 30
```

ORDER BY

Digunakan untuk mengurutkan data hasil dari *query* yang dilakukan. Dapat diurutkan secara naik (ASC, yang merupakan *default*-nya), atau secara menurun (DESC). Contoh berikut akan mendapatkan data buku yang judulnya ada *string* PHP dan akan diurutkan secara menurun (desc) menurut penerbitannya:

```
SELECT id, title, published FROM document where title LIKE '%PHP'
    ORDER BY published;
```

LIMIT

Statement ini akan membatasi baris yang akan keluar dari hasil *query* yang dilakukan. Contoh berikut akan menampilkan sepuluh data pertama dari tabel document:

```
SELECT titles, published FROM document LIMIT 10;
```

Dan contoh berikut akan mengembalikan sepuluh baris berikutnya (baris 11-20):

```
SELECT titles, published FROM document LIMIT 10, 10;
```

PROCEDURE

Didalam MySQL kita bisa mendefinisikan atau membuat prosedur dalam bahasa C++ yang dapat mengakses dan merubah data di dalam sebuah query sebelum data itu di kirim kembali ke client.

Modul VI
PHP dan MySQL

Aplikasi datababase pada situs website diperlukan untuk mengorganisir informasi yang banyak agar memudahkan pengunjung untuk mencari informasi yang mereka butuhkan.

PHP telah menyediakan fasilitas koneksi untuk hampir semua program database populer baik yang komersial maupun gratis. Berikut ini contoh membangun database di situs web dengan menggunakan MYSQL. MYSQL adalah salah satu program database yang gratis dan cukup handal.

Akses Database

Secara umum akses ke database melalui tiga tahapan:
1. koneksi ke database (persiapan)
2. query/permintaan data (operasi)
3. pemutusan koneksi

Koneksi ke database dilakukan menggunakan fungsi-fungsi *mysql_connect(), mysql_pconnect(), mysql_select_db()*

Mysql_connect()

Mysql_connect() digunakan untuk melakukan koneksi ke program database MYSQL. Sintaksnya:

mysql_connect(nama_host,nama_user,password)

Jika parameter nama host tidak dideklarasikan, otomatis akan berisi localhost. Koneksi ke database akan secara otomatis terputus pada saat script program selesai dieksekusi seluruhnya, kecuali diberikan perintah mysql_close(). Fungsi mysql_connect() akan menghasilkan nilai true jika koneksi berhasil dilakukan dan nilai fase jika gagal.

Contoh :

```
<?
//contoh koneksi database MYSQL
//connect.php
$sambung = mysql_connect(“localhost”,”mahasiswa”,”rahasia”);
if($sambung){
    echo (“Koneksi Berhasil”);
}else {
    echo (“Koneksi Gagal”);
}
?>
```

Pada contoh diatas, mysql_connect() diberi nilai host = “localhost”, namauser = “mahasiswa” dan password = “rahasia”. Bila koneksi berhasil dilakukan dilayar akan muncul pesan Koneksi Berhasil, jika gagal maka akan muncul pesan Koneksi Gagal.

Fungsi mysql_pconnect() juga digunakan untuk membangun akses ke database, sama dengan fungsi mysql_connect(). Sedikit perbedaan adalah jika menggunakan fungsi mysql_pconnect(), koneksi tidak akan terputus meskipun program telah selesai dieksekusi.

Membuat Database

Fungsi mysql_create_db() digunakan untuk membuat sebuah database. Biasanya untuk sebuah aplikasi digunakan sebuah database. Sebuah database akan berisi beberapa tabel. Sintaks fungsi mysql_create_dn() sebagai berikut :

Mysql_create_db(“database”)

Contoh :

```
<?
//contoh membuat database MYSQL
//create.php
$sambung = mysql_connect(“localhost”,”mahasiswa”,”rahasia”);
if($sambung){
     echo (“Koneksi Berhasil”);
}else {
     echo (“Koneksi Gagal”);
}
$buat = mysql_create_db(“data_mahasiswa”);
if($buat){
     echo (“<br><br>Database data_mahasiswa berhasil dibuat”);
}else {
echo (“<br><br>Database data_mahasiswa gagal dibuat”);
}
?>
```

Membuat Tabel

Database memuat informasi yang diatur dalam tabel-tabel. Misalnya, sebuah database berita akan terdiri atas tabel berita politik, berita olahraga, berita teknologi dan lain-lain. Sebuah tabel mengandung field-field data, contohnya tabel berita politik berisi data dengan field judul berita, isi berita dan nama wartawan.

PHP tidak menyediakan fungsi khusus untuk membuat tabel dengan field-fieldnya, sehingga untuk membuat tabel data tetap menggunakan sintaks dari program database (MYSQL) yang digunakan, kemudian sintaks ini dioperasikan menggunakan fungsi mysql_query(). Sebagai contoh, dibuat sebuah database mahasiswa sebuah universitas, table data pribadi dengan field-field nomor induk mahasiswa, nama mahasiswa dan alamat mahasiswa.

```
<?
//contoh membuat tabel database MYSQL
//create_tabel.php
$sambung = mysql_connect(“localhost”,”mahasiswa”,”rahasia”);
if($sambung){
     echo (“Koneksi Berhasil”);
}else {
     echo (“Koneksi Gagal”);
}
$buat = mysql_create_db(“data_mahasiswa”);
if($buat){
     echo (“<br><br>Database data_mahasiswa berhasil dibuat”);
}else {
echo (“<br><br>Database data_mahasiswa gagal dibuat”);
}
$perintah1= “CREATE TABLE data_pribadi
        (  nim int(10) AUTO_INCREMENT PRIMARY KEY,
           nama char(100),
           alamat varchar(255) )”;
$buat_tabel=mysql_db_query(“data_mahasiswa”, $perintah1);
if($buat_tabel){
     echo (“<br>Tabel data pribadi berhasil dibuat”);
}else {      echo (“<br>Tabel data pribadi gagal dibuat”);
```

```
}
?>
```

Jika create tabel berhasil dibuat, maka tabel data_pribadi terdiri dari 3 field, yaitu nim, nama dan alamat.

Tipe Data Field

Data yang terdapat dalam tabel berupa field-field yang berisi nilai dari data tersebut. Nilai data dalam field ini memiliki tipe sendiri-sendiri. Contoh, field nim bertipe integer dengan lebar field 10, field nama bertipe character dengan lebar field 100, dan field alamat bertipe varchar dengan lebar field maksimum 255.

MYSQL mengenal beberapa tipe data field, yaitu:

- **Tipe data Numerik**

Tipe data Numerik dibedakan dalam dua macam kelompok, tipe data integer dan tipe data floating point. Tipe data integer untuk data bilangan bulat, tipe data floating point untuk bilangan desimal. Yang termasuk tipe data numerik adalah : TINYINT, SMALLINT, MEDIUMINT, INT, BIGINT, FLOAT(x), FLOAT dan DOUBLE

- **Tipe Data String**

Yang termasuk dalam tipe data String adalah tipe-tipe data CHAR, VARCHAR, TINYBLOB, TINYTEXT, BLOB, TEXT, MEDIUMBLOB, MEDIUMTEXT. LONGBLOB, LONGTEXT, ENUM('elemen 1','elemen2',…), SET('elemen 1','elemen2',…)

Tipe Data char() dan varchar()

Type data char() dan varchar() pada prinsipnya sama, perbedaanya pada jumlah memori yang dibutuhkan untuk penyimpanan. Memori penyimpanan yang dibutuhkan untuk tipe data char() bersifat statis, besarnya bergantung pada beberapa jumlah character yang ditetapkan pada saat field tersebut dideklarasikan. Sebaliknya, tipe data varchar() besarnya memori penyimpanan bergantung pada berapa character yang dipakai. Untuk lebih jelasnya pada contoh dibawah ini:

Nilai	**Char(4)**	**Memori penyimpanan**	**Vachar(4)**	**Memori penyimpanan**
''	' '	4 bytes	''	1 byte
'ab'	'ab'	4 bytes	'ab'	3 bytes
'abcd'	'abcd'	4 bytes	'abcd'	5 bytes
'abcdefgh'	'abcdefgh'	4 bytes	'abcdefgh'	5 bytes

Tipe Data Tanggal

Untuk data tanggal dan waktu(jam) tersedia tipe-tipe data field sebagai berikut dengan besarnya kebutuhan memori penyimpanan untuk masing-masing tipe data string sebagai berikut:

Tipe data	Kisaran Nilai	Kebutuhan memori penyimpanan
DATETIME	'1000-01-01 00:00:00' to '9999-12-31 13:59:59'	3 bytes
DATE	'1000-01-01' to '9999-12-31'	8 bytes
TIMESTAMP	'1970-01-01 00:00:00' – '2037'	4 bytes
TIME	'-838:59:59' to '838:59:59'	3 bytes
YEAR	1901 – 2155	1 byte

Tahap Operasi

Setelah melakukan koneksi ke database, membuat sebuah database dan sebuah tabel beserta field-fieldnya, tahap persiapan telah selesai. Tahap berikutnya, yaitu tahap operasi di mana

dilakukan operasi-operasi memasukkan data, mengambil data, mengedit data, menghapus data, dan lain-lain, dapat dimulai.

PHP tidak menyediakan fungsi-fungsi khusus untuk operasi data, sehingga sintaks yang dipakai adalah sintaks perintah-perintah MYSQL. Untuk melakukan operasi-operasi data menggunakan PHP ururtannya sebagai berikut:

1. Koneksi ke database
2. Deklarasikan sebuah variabel string yang berisi sintaks perintah MYSQL yang akan dilakukan
3. Laksanakan sintaks MYSQL menggunakan fungsi mysql_query(), jika sintaks yang akan dijalankan menghasilkan output yang akan ditampilkan, deklarasikan sebuah variabel untuk menampung hasil tersebut.
4. Mengambil hasil dari sintaks MYSQL yang dilaksanakan menggunakan fungsi-fungsi mysql_fetch_array(), mysql_fetch_row(), mysql_num_rows, dan lain-lain. Bergantung pada format output yang diinginkan.

Memasukkan Data

Untuk memasukkan data ke database, digunakan sintaks:
INSERT INTO nama_table (field1, field2, …) VALUES (‘data1’, ‘data2’,…)
Jika parameter field tidak dideklarasikan, data yang dimasukkan jumlahnya harus sama dengan jumlah field dari table tersebut.

Contoh:

```
<?
//contoh memasukkan data
//datamasuk.php
mysql_connect(“localhost”,”mahasiswa”,”rahasia”);
mysql_select_db(“data_mahasiswa”);
$perintah= “INSERT INTO data_pribadi
                (nama, alamat)
              VALUES
                (‘Budi’,’Jl.Ganesa no.10 Bandung’)”; // perintah masukkan data

$hasil=mysql_query($perintah);
if($hasil){
    echo (“Input data berhasil”);
}else {
    echo (“Input data gagal”);
}
?>
```

Agar proses input data ini lebih ‘user friendly’, dapat dibuat sebuah form HTML untuk input data, contoh:

```
<HTML>
<HEAD>
     <TITLE> Form Input Data </TITLE>
</HEAD>

<BODY>
<CENTER>
<TABLE border = 1>
<tr>
<td align = center>Form Input Data Mahasiswa</td>
</tr>
<tr>
```

```
<td>
    <table>
        <form method=post action=input.php>
        <tr>
            <td>Nama</td><td><input type = text name=nama
size=20></td>
         </tr>
        <tr>
            <td>Alamat</td><td><input type = text name=alamat
size=40></td>
        </tr>
        <tr>
           <td> </td>
            <td align=right><input type =submit name=submit
value=Kirim></td>
        </tr>
     </table>
</td>
</tr>
</TABLE>
</CENTER>
</BODY>
</HTML>
```

Form HTML ini akan mengirimkan dua variabel yaitu variabel $nama dan $alamat ke file input.php, sesuai yang tertera di parameter ACTION dan FORM HTML

Berikut ini file input.php

```
<?
//contoh memasukkan data dengan variabel dari form HTML
//input.php
mysql_connect("localhost","mahasiswa","rahasia");
mysql_select_db("data_mahasiswa");
$perintah= "INSERT INTO data_pribadi
                (nama, alamat)
              VALUES
                ('$nama','$alamat')"; // perintah masukkan data

$hasil=mysql_query($perintah);
if($hasil){
     echo ("Input data berhasil");
}else {
     echo ("Input data gagal");
}
?>
```

File input.php akan mengolah variabel tersebut dan memasukkannya ke dalam database. Dalam sintaks INSERT yang dideklarasikan hanya field nama dan alamat, sedangkan field nim tidak perlu diisikan karena field tersebut memiliki atribut AUTO_INCREMENT yang artinya MYSQL akana mengenerate sendiri data untuk field tersebut.

Mencari Data

Untuk mencari satu atau lebih data di database digunakan sintaks berikut:

SELECT field1,field2,… FROM nama_table WHERE syarat1, syarat2,… ORDER BY nama_field

Jika field yang dicari adalah semua field dari tabel maka tidak perlu nama field dideklarasikan semua tetapi cukup digantikan dengan tanda * (asteriks) maka semua field akan diakses. Paramater ORDER BY menunjukkan data akan diurutkan berdasar field yang mana, default urutan adalah dari yang terkecil (urutan angka), dari A – Z (urutan huruf) dan dari data yang pertama dimasukkan ke data yang terakhir (urutan waktu). Urutan ini dapat dibalik dengan menambahkan atribut DESC. Sebagai contoh akan dicari semua data pada database data_mahasiswa, table data_pribadi untuk ditampilkan urut berdasarkan field nama.

```
<HTML>
<HEAD>
     <TITLE>Mencari data di database</TITLE>
</HEAD>

<BODY>
<TABLE>
<tr>
<td align = center> Data Mahasiswa</td>
</tr>
<tr>
<td>
    <table border = 1>
         <tr>
            <td>Nama</td><td>NIM</td><td>ALAMAT</td>
        </tr>
<?
//contoh mencari data dengan urutan berdasarkan nama
//cari.php
mysql_connect(“localhost”,”mahasiswa”,”rahasia”);
mysql_select_db(“data_mahasiswa”);

// perintah mencari data di deklarasikan dalam variabel perintah
$perintah= “SELECT * FROM data_pribadi ORDER BY nama”;
$hasil=mysql_query($perintah); //perintah dilaksanakan hasil disimpan
dlm $hasil

while($data=mysql_fetch_row($hasil)){
  echo      (“<TR>      <TD>$data[1]</TD>      <TD>$data[0]</TD>
<TD>$data[2]</TD>
            </TR>”);
}
?>
    </table>
</td>
</tr>
</TABLE>
</CENTER>
</BODY>
</HTML>
```

Data yang dicari disimpan dalam variabel $hasil, variabel ini bertipe array dengan urutan default dari A ke Z, jika data akan diurutkan kebalikannya (dari Z ke A) ditambahkan atribut DESC, contoh program diatas, sintaks mencari data diubah menjadi :

$perintah= “SELECT * FROM data_pribadi ORDER BY nama DESC”;

Fungsi mysql_fetch_row()

Pada contoh program diatas, hasil dari query ke database disimpan dalam sebuah variabel yaitu $hasil. Untuk mengambil isi dari variabel ini digunakan fungsi mysql_fetch_row(). Fungsi mysql_fetch_row() mengambil data dari variabel $hasil secara baris perbaris. Pengambilan pertama adalah baris data yang paling atas. Data yang akan diambil dalam bentuk array, dimana elemen array adalah field-field dari tabel data. Pembacaan dilakukan perbaris data, sehingga perintah while akan mendapatkan nilai false, dengan demikian perulangan dihentikan.

Fungsi mysql_fetch_array()

Selain menggunakan fungsi mysql_fetch_row() untuk mengambil hasil query ke database, dapat juga digunakan fungsi mysql_fetch_array(). Fungsi ini hampir sama, dimana data dibaca baris perbaris, perbedaannya menggunakan mysql_fetch_array() hasil yang diperoleh dalam bentuk array assosiatif. Sehingga bila pada pembacaan mysql_fetch_row indeksnya adalah numeric, maka pada hasil pembacaan dengan mysql_fetch_array($hasil) pembacaan menghasilkan array dengan indeks string sesuai dengan nama fieldnya, contoh : $row[nama], $row[nama],$row[alamat]. Pembacaan dilakukan baris perbaris data, sampai tidak ada lagi baris data yang dibaca.

Mengedit Data

Untuk mengedit data menggunakan sintaks berikut:

UPDATE nama_table SET field1=nilai_baru, field2=nilai_baru,… WHERE syarat1, syarat2

Sebagai contoh, akan dicoba untuk mengedit salah satu data dari tabel data_mahasiswa.

```
<?
//file include untuk koneksi database
//db.inc.php
mysql_connect(“localhost”,”mahasiswa”,”rahasia”);//koneksi database
mysql_select_db(“data_mahasiswa”);
?>

<?
//form untuk memilih data yang akan diedit
//edit.php
?>
<HTML>
<HEAD>
      <TITLE>Edit data</TITLE>
</HEAD>

<BODY>
<TABLE>
<tr>
<td align = center> Edit Data </td>
</tr>
<tr>
<td>
    <table border = 1>
<?
include “db.inc.php”; //koneksi ke database
$perintah= “SELECT * FROM data_pribadi”;
$hasil=mysql_query($perintah); //perintah dilaksanakan hasil disimpan
dlm $hasil
```

```
while($row=mysql_fetch_array($hasil)){
    echo (“<TR> <TD>$row[nama]</TD>”);
    echo (“ <TD>$row[nim]</TD>”);
    echo (“ <TD>$row[alamat]</TD>”);
    echo (“<td> <a href=\”edit_form.php?id=$row[nim]\”> Edit
</a></td></TR>”);
}
?>
</table>
</td></tr></TABLE>
</BODY>
</HTML>
```

Untuk contoh program mengedit data ini, terdiri atas empat buah file program. File db.inc.php adalah file untuk koneksi database, sehingga untuk koneksi ke database untuk tiga file lainnya cukup dilakukan dengan menginclude file db.inc.php ini. Dengan mendeklarasikan koneksi database ke dalam sebuah file include tersendiri akan memudahkan. Jika suatu saat perlu dilakukan perubahan user_name atau password untuk koneksi database. Untuk penamaan perlu dilakukan hati-hati, tetaplah konsisten menggunakan ekstensi.php agar isi file tersebut tidak dapat terlihat. Kehati-hatian ini sangat diperlukan mengingat isi file koneksi ini sangat penting untuk keamanan data dan bersifat rahasia.

File kedua untuk contoh program mengedit data adalah file edit.php, file ini akan menampilkan keseluruhan data yang ada dalam tabel, kemudian pada kolom terakhir akan tampil menu edit yang jika di klik akan membawa program untuk menjalankan file edit_form.php

File ketiga contoh program mengedit data adalah file edit_form.php. File ini akan menampilkan form untuk mengedit data yang telah dipilih pada form sebelumnya. Mekanismenya adalah pada form yang pertama (file edit.php) pengguna memili salah satu data yang akan diedit dengan mengklik menu edit pada kolom yang paling kanan. Pada saat menu edit ini diklik program akan menuju file edit_form.php dan mengirimkan variael $id yang isinya nomor karyawan yang dipilih untuk diedit datanya. Berikut file program edit_form.php

```
</HEAD>

<BODY>
<CENTER>
<TABLE border = 1>
<tr>
<td align = center>Form Edit Data Mahasiswa</td>
</tr>
<tr>
<td>
    <table>
<?
include "db.inc.php"; //koneksi ke database
$perintah= "SELECT * FROM data_pribadi WHERE nim='$id'";
$hasil=mysql_query($perintah);
$row = mysql_fetch_array($hasil);
?>
        <form method=post action=edit_data.php>
        <tr>
            <input type=hidden name="id" value ="<?echo
"$row[nim]"?>" >
             <td>Nama</td><td><input type = text name=nama size=20
value =
```

```
                    " <? echo "$row[nama]"?>"  ></td>
          </tr>
        <tr>
            <td>Alamat</td><td><input type = text name=alamat size=40
value =
                    " <? echo "$row[alamat]"?>"  ></td>
        </tr>
        <tr>
           <td> </td>
            <td     align=right><input     type     =submit     name=submit
value=Edit></td>
        </tr>
     </table>
</td>
</tr>
</TABLE>
</CENTER>
</BODY>
</HTML>
```

Editlah data yang akan diubah, misalnya mengubah alamat mahasiswa yang baru pindah rumah. Isilah alamat yang baru, kemudian klik tombol edit.

Dengan mengklik tombol edit, maka program akan menuju ke file program yang ke empat, yaitu file edit_data.php dengan membawa tiga variabel, yaitu variabel $id, yang berisi data nomor induk mahasiswa (nim), variabel $nama yang berisi data nama mahasiswa dan variabel $alamat yang berisi data alamat. Pada file ini dilakukan perubahan data pada database dengan menggunakan sintaks :

$perintah = "UPDATE data_pribadi SET nama='$nama', alamat = '$alamat' where nim = '$id'";

Untuk melihat hasilnya, apakah data sudah berubah atau belum, program diarahkan kembali ke file edit.php dengan perintah header("location:edit.php"). File program edit_data.php sebagai berikut:

```
<?
//edit_data.php
include"db.inc.php";
$perintah = "UPDATE data_pribadi SET nama='$nama', alamat =
'$alamat' where
                nim = '$id'";
mysql_query($perintah);
header("location:edit.php");
?>
```

Menghapus Data

Untuk menghapus data, menggunakan sintaks berikut:

DELETE FROM nama_table WHERE syarat1,syarat2,…

Sebagai contoh, form edit data pada program diatas ditambahkan menu untuk menghapus data sebagai berikut:

echo ("<td> Delete </td></TR>");

Menu Delete akan membawa program ke file delete.php yang isinya untuk mengapus data yang telah dipilih pada form di atas. File delete.php sebagai berikut:

```
<?
//delete.php
include”db.inc.php”;
$perintah = “DELETE FROM data_pribadi WHERE nim=’$id’”;
mysql_query($perintah);
header(“location:edit.php”);
?>
```

Menambah, Mengedit dan Menghapus Field Tabel

Menambah field table menggunakan sintaks berikut :

ALTER TABLE nama_table ADD nama_field tipe_field atribut_field

Contohnya, pada table mahasiswa akan ditambahkan field “jurusan” dengan tipe “varchar” panjang field 50 character.

```
<?
//contoh menambah field tabel database MYSQL
//tambah_field.php
mysql_connect(“localhost”,”mahasiswa”,”rahasia”);
mysql_select_db(“data_mahasiswa”);
$perintah= “ALTER TABLE data_pribadi ADD jurusan varchar(255)”;
$tambah_field=mysql_db_query(“data_mahasiswa”,$perintah);
if($tambah_field){
     echo (“<BR><BR>Field jurusan berhasil ditambahkan”);
}else {
echo (“<BR><BR>Field jurusan gagal ditambahkan”);
}
?>
```

Jalankan program diatas di browser, Jika berhasil akan tampil dilayar tulisan:

Field jurusan berhasil ditambahkan

Struktur tabel data_mahasiswa akan berubah menjadi 4 field, yaitu nim yang bertipe integer, nama bertipe varchar(100), alamat bertipe varchar(255) dan jurusan bertipe varchar(255).

Suatu saat, adakalanya tabel yang telah dibuat memerlukan modifikasi atau perubahan, misalnya tipe field yang sebelumnya char ingin diubah menjadi varchar, atau panjang field yang semula char(50), misalnya ingin diperlebar menjadi char(100), Untuk mengubah field data ini menggunakan sintaks :

ALTER TABLE nama_tabel MODIFY nama_field tipe_field atribut_field

Sebagai contoh field jurusan pada tabel data_pribadi akan diubah tipe fieldnya menjadi char(50)

```
<?
//contoh mengubah field tabel database MYSQL
//ubah_field.php
```

```
mysql_connect(“localhost”,”mahasiswa”,”rahasia”);
mysql_select_db(“data_mahasiswa”);
$perintah= “ALTER TABLE data_pribadi MODIFY jurusan char(50)”;
$ubah_field=mysql_db_query(“data_mahasiswa”,$perintah);
if($ubah_field){
    echo (“<BR><BR>Field jurusan berhasil diubah”);
}else {
    echo (“<BR><BR>Field jurusan gagal diubah”);
}
?>
```

Jalankan di browser, jika berhasil akan muncul dilayar tulisan berikut:

Field jurusan berhasi diubah

Struktur databasenya akan berubah juga, dimana field jurusan mempunyai tipe data char(50).

Untuk menghapus sebuah field dari tabel menggunakan sintaks berikut:

ALTER TABLE nama_tabel DROP nama_field tipe_field

Sebagai contoh, field jurusan pada tale data_pribadi akan dihapus, menggunakan program berikut:

```
<?
//contoh menghapus field tabel database MYSQL
//hapus_field.php
mysql_connect(“localhost”,”mahasiswa”,”rahasia”);
mysql_select_db(“data_mahasiswa”);
$perintah= “ALTER TABLE data_pribadi DROP jurusan”;
$hapus_field=mysql_db_query(“data_mahasiswa”,$perintah);
if($ubah_field){
    echo (“<BR><BR>Field jurusan berhasil dihapus”);
}else {
    echo (“<BR><BR>Field jurusan gagal dihapus”);
}
?>
```

Jika program berhasil menghapus field, maka dilayar akan tampil tulisan :

Field jurusan berhasil dihapus

Untuk membantu melihat struktur basis data dapat digunakan aplikasi MYSQLadmin yang juga dibuat menggunakan bahasa pemrograman PHP.

Modul VII

OOP, Cookies, Session Handling dan Mengirim E-mail

Object Oriented Programming

PHP pada dasarnya tidak dimaksudkan untuk pemrograman berorientasi objek, akan tetapi PHP menyediakan fungsi untuk membuat sebuah class. Kemampuan PHP utuk permrogaraman berorientasi objek sangat minim. Pada bagian ini dibahas poin-poin yang dapat dilakukan menggunakan PHP

Class dan Object

Object : representasi sesuatu di dalam dunia nyata, bisa berupa benda, aktivitas dan apa saja.
Class : blue print dari sebuah object, yang mendefinisikan implementasi detail dari objek.

Object – object yang sama berada pada satu class. Sebuah object dibuat berdasarkan definisi class dari object tersebut, contoh Toyota Kijang adalah sebuah object yang dibuat berdasarkan definisi dari class mobil.

Dalam PHP, class merupakan kumpulan variabel dan fungsi yang bekerja dengan variabel tersebut, dan object adalah instans atau turunan dari class. Sebuah object akan mewarisi variabel dan fungsi dari class tempatnya berasal.

Untuk mendeklarasikan sebuah class digunakan perintah:

```
Class nama_class{
}
```

Untuk membuat sebuah object, digunakan perintah :

```
$sebuah_objek = new nama_class;
```

Untuk mengakses fungsi-fungsi dari class digunakan operator → .

Misalkan akan dibuat sebuah class yang mendefinisikan sebuah kotak. Class kotak akan memiliki variabel-variabel yang menggambarkan karakteristik dari sebuah kotak dan fungsi untuk menggambar kotak.

```
<?
//classkotak.php

class Kotak{
   //karakteristik kotak
    var $x_awal = 0;
    var $y_awal = 0;
   var $panjang= 250;
   var $lebar = 50;

  //fungsi-fungsi (method)
  //fungsi set koordinat x awal
   function set_x_awal($x_awal){
        $this -> x_awal =  $x_awall;
   }
   //fungsi set koordinat y awal
   function set_y_awal($y_awal){
        $this -> y_awal =  $y_awal;
   }
   //fungsi set panjang kotak
   function set_panjang($panjang){
        $this -> panjang =  $panjang;
```

```
  }
//fungsi set lebar kotak
  function set_lebar($lebar){
      $this -> lebar =  $lebar;
  }
//fungsi menggambar kotak
  function gambar_kotak($x_awal, $y_awal, $panjang, $lebar){
      //kirim header
       Header("Content-Type: image.jpeg");

      //deklarasi variabel image
      $img = ImageCreate(500,500);

       //delarasi variabel warna
      $putih = ImageColorAllocate($img,255,255,255);
      $hitam = ImageColorAllocate($img,0,0,0);

       //membuat kotak
       ImageFill($img,0,0,$putih);
       Imagerectangle($img,$x_awal,$y_awal,$panjang,$lebar,$hitam);

      //output image ke browser
       ImageJPEG($img);

  }
}
?>
```

Class kotak mendefinisikan detail dari sebuah kotak. Karakteristik dari kotak adalah koordinat awal dimana kotak akan digambarkan, pnjang kotak dan lebar kotak. Untuk membuat sebuah kotak, maka dibuat sebuah object yang merupakan instans dari class kotak.

```
<?
//contoh membuat object dari sebuah class
//objectkotak.php

//mengaitkan dengan file class
include”classkotak.php”;

//membuat sebuah object kotak1
$kotak1 = new Kotak;

//mengakses fungsi-fungsi class
$x_awal = $kotak1 → x_awal;
$y_awal = $kotak1 → y_awal;
$panjang= $kotak1 → panjang;
$lebar= $kotak1 → lebar;
$kotak1→gambar_kotak($x_awal, $y_awal,$panjang, $lebar);
?>
```

Kotak yang dihasilkan adalah persis dengan kotak yang didefinisikan oleh class Kotak. Dalam pemrograman berorientasi object, setelah class dibuat, maka dapat dibuat banyak object yang dapat persis seperti class-nya atau dapat juga mempunyai karakteristik yang berbeda dengan classnya (untuk kotak, ukuran dapat berbeda-beda).

Contoh berikut membuat object kotak kedua dengan ukuran yang berbeda

```
<?
//contoh membuat object dari sebuah class
//objectkotakkedua.php

//mengaitkan dengan file class
include"classkotak.php";

//membuat sebuah object kotak yang kedua
$kotak2 = new Kotak;

//definisi karakteristik baru
$kotak2→set_x_awal(20);
$kotak2→set_y_awal(20);
$kotak2→panjang(300);
$kotak2→lebar(300);


//mengakses fungsi-fungsi class
$x_awal = $kotak2 → x_awal;
$y_awal = $kotak2 → y_awal;
$panjang= $kotak2 → panjang;
$lebar= $kotak2 → lebar;
$kotak2→gambar_kotak($x_awal, $y_awal,$panjang, $lebar);
?>
```

Cookies

Cookies adalah mekanisme penyimpanan sebuah variabel data pada client browser. Untuk mendeklarasikan sebuah cookies menggunakan fungsi setcookie(). Cookies adalah bagian dari HTTP header, sehingga cookies harus dideklarasikan sebelum program mengirimkan output apapun ke client browser.

Contoh :

```
<?
//contoh cookies
//cookies.php
setcookie("test_cookies","adalah variabel cookies");
?>
<html>
<head><title>Cookies</title></head>
<body>
<?
Echo("<A HREF=\"cookies.php\">Klik disini !!!</a>");
?>
</body>
</html>
```

Dalam program cookies.php diatas, telah ditetapkan sebuah cookies dengan nama test_cookies, dengan nilai sebuah string "adalah variabel cookies". Jika menu "Klik disini !!!" di klik, program akan berakhir. Kemudian program tersebut akan dieksekusi akan tetapi variabel $test_cookies akan tetap ada di komputer pemakai.

Untuk menguji apakah ada cookies atau tidak dapat dilihat pada contoh berikut

```
<?
```

```
//contoh cookies
//issetcookies.php
if(isset($test_cookies)){
    echo("\$test_cookies ".$HTTP_COOKIE_VARS["test_cookies"]);
}else {
    echo("Variabel cookies belum diset");
}
?>
```

Program cookies.php bila diuji dengan program issetcookies.php akan menghasilkan nilai True. Dilayar akan tampil hasil sebagai berikut :

$test_cookies adalah variabel cookies

Eksistensi cookies dapat diatur dengan atribut “expire”. Contoh jika diinginkan $test_cookies berumur 1 jam (3600 detik), sintaksnya adalah sebagai berikut:

Setcookie(“test_cookies”,”adalah variabel cookies”, time()+3600);

Session Handling

Fasilitas session adalah salah satu keunggulan PHP, menggunakan session, seorang pengunjung situs web dapat memiliki variabel yang akan terus ada selama dalam kunjungannya.

Variabel Session

Variabel session adalah sebuah variabel global, yang akan dibuat pada saat session dimulai. Untuk memulai session dapat secara explisit dengan fungsi session_start() atau secara implisit menggunakan fungsi session_register(). Untuk lebih jelasnya dalam contoh berikut:

```
<?
//contoh variabel session
//session1.php

session_start();
$test_session = "ini variabel session";
session_register("test_session");
echo("<a href=session2.php>LANJUT>>><a\>");
?>

<?
//contoh variabel session
//session2.php
session_start();
echo"\$test_session = $test_session";
?>
```

Jalankan program session1.php, klik “LANJUT>>>” , program akan menjalankan session2.php, hasilnya dilayar akan muncul tulisan sebagai berikut:

$test_session = ini variabel session

Aplikasi Session

Session biasa digunakan untuk aplikasi aplikasi keamanan (autentikasi) dan e-commerce (*online shopping*). Sebagai contoh, sebuah aplikasi autentikasi. Seorang pengunjung halaman situs akan diperiksa terlebih dahulu apakah dia boleh memasuki situs tersebut. Misalnya saja pengunjung yang mendapatkan izin diberi identifikasi “user”. Setiap user juga tidak dapat memasuki semua halaman situs, sebab ada halaman yang hanya boleh dimasuki oleh “super user”.

Mengirim Email

PHP menyediakan fasilitas untuk mengirim email, menggunakan fungsi mail(), sintaksnya sebagai berikut:

Mail(tujuan, hal, pesan, additional_headers);

Contoh program kirim email :

```
<?
//email.php
if($submit) {
        mail($tujuan, $hal, $pesan, "From:
webmaster@$SERVER_NAME");
        echo("<br>Email telah dikirimkan ke $tujuan");
}
?>

<HTML>
<HEAD>
<TITLE> Kirim Email </TITLE>
</HEAD>
<BODY>
<? Echo ("<FORM METHOD = post ACTION=$PHP_SELF>") ?>
<h3> Kirim Email </h3>
<p> To: <INPUT TYPE = text NAME = tujuan SIZE=30>
<br> Subject: <INPUT TYPE = text NAME = hal SIZE=30>
<P> Message : <br> <TEXTAREA NAME="pesan" ROWS="5"
COLS="70"></TEXTAREA>
<P><INPUT TYPE="SUBMIT" NAME="submit" VALUE= "Kirim" SIZE
= 30>
</FORM>
</BODY>
</HTML>
```

Cobalah simpan program diatas dengan nama email.php kemudian cobalah di browser, cobakan untuk mengirimkan email.

Modul VIII

Pemrograman PHP menggunakan Template

1. Konsep Template

Konsep template muncul dilatarbelakangi oleh kurang efisiennya pemrograman PHP yang konvensional (*embedded* HTML), terutama dalam pembuatan program dalam skala besar. Pada pemrograman konvensional PHP di-*embed*-kan dalam HTML untuk membuat tampilan web yang dinamis. *Web programmer* (pembuat script PHP) akan menambahkan programnya (*script* PHP) dalam web yang telah didisain oleh *web designer*, sedemikian rupa sehingga tampilan web menjadi dinamis. Namun jika ada perubahan dalam disain web (misalnya perubahan tata letak tanggal) maka *web programmer* juga harus merubah posisi *script* yang digunakan untuk menampilkan tanggal. Untuk web skala kecil maka hal tersebut bukan menjadi masalah, karena perubahan yang dilakukan sedikit. Akan tetapi untuk web dalam skala besar tentunya akan banyak perubahan yang dilakukan, hal ini tentunya menjadi tidak efisien. Salah satu cara untuk mengatasi hal tersebut adalah penggunaan template. Dengan menggunakan template maka perubahan-perubahan hanya dilakukan pada file template saja. Sedangkan file lain (yang mengacu pada template tersebut tidak perlu diubah).

Selain itu penggunaan template memungkinkan *web programmer* dan *web designer* bekerja secara parallel untuk membuat suatu web yang dinamis. Namun tetap diperlukan adanya kesepakatan tentang kode-kode (tag-tag) yang digunakan untuk memproses file template. Dengan demikian web designer dapat berkonsntrasi dalam pembuatan design, sedangkan web programmer dapat berkonsentrasi dalam pembuatan script PHP.

Ide dasar dari konsep template ini adalah file template yang dibuat oleh web designer diberi kode-kode (tag-tag) khusus, yang nantinya akan digantikan oleh hasil pemrosesan script yang dibuat oleh web programmer. File template di-*parsing* kemudian tag-tag yang ada digantikan dengan keluaran dari fungsi yang memproses script.

2. Teknik pemrograman menggunakan template

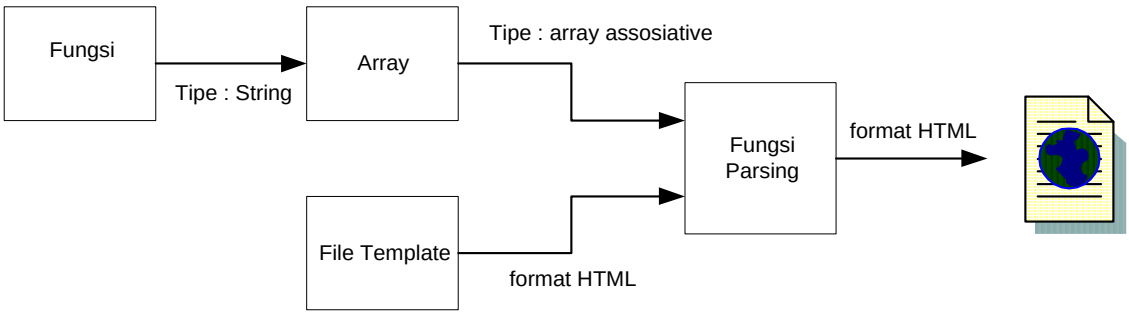
Teknik pemrograman template ini telah banyak dikembangkan oleh programmer PHP, terutama untuk pembuatan web dalam skala besar. Ada beberapa tool yang telah dibuat untuk pemrograman menggunakan template, diantaranya FastTemplate, EasyTemplate, Smarty dan lain-lain. Dalam pelatihan kali ini akan dibahas tool yang sederhana yang digunakan untuk pemrograman template.

Sebelum membahas lebih lanjut tentang teknik pemrograman menggunakan template, terlebih dahulu akan dijelaskan beberapa istilah yang penting.

a. File template adalah file yang berisi halaman template HTML dengan tag-tag yang nanti akan di ganti dengan keluaran dari sebuah fungsi. Fungsi tag-tag tersebut adalah untuk menandai bagian-bagian dalam file template yang nantinya akan diganti dengan keluaran fungsi.
b. Fungsi adalah kumpulan script baik PHP maupun HTML yang melaksanakan suatu proses atau logika tertentu. Fungsi tersebut bisa mempunyai masukan (fungsi) atau tidak (prosedur). Fungsi tersebut akan mengembalikan keluaran dalam format *string*.
c. Fungsi *parsing* fungsi ini merupakan fungsi yang mem-*parsing* file template, kemudian mencari tag, jika ditemukan tag tersebut diganti dengan keluaran fungsi. Fungsi *parsing* mempunyai dua buah masukan yaitu file template dan array yang berisi tag-tag. Array ini digunakan untuk menyimpan pengembalian dari fungsi. Untuk memanggil fungsi parsing digunakan perintah berikut :

```
<?
    Parse ([nama dan path file template],[tag array associative]);
?>
```

Untuk lebih jelasnya dapat dilihat dalam gambar berikut :



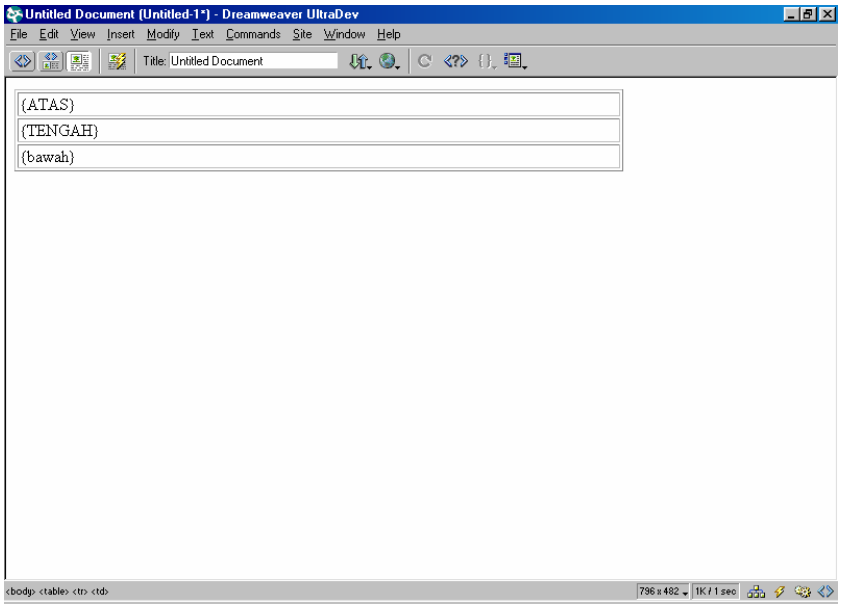
Gambar 8.1 Proses dalam pemrograman template

Setelah mengetahui proses yang terjadi dalam pemrograman template, berikut akan dijelaskan tentang teknik pemrograman menggunakan template :

a. Pembuatan File Template
File template merupakan file yang isinya dalam format HTML dan disertai tag-tag tertentu. Format penulisan tag-tag tersebut adalah sebagai berikut :

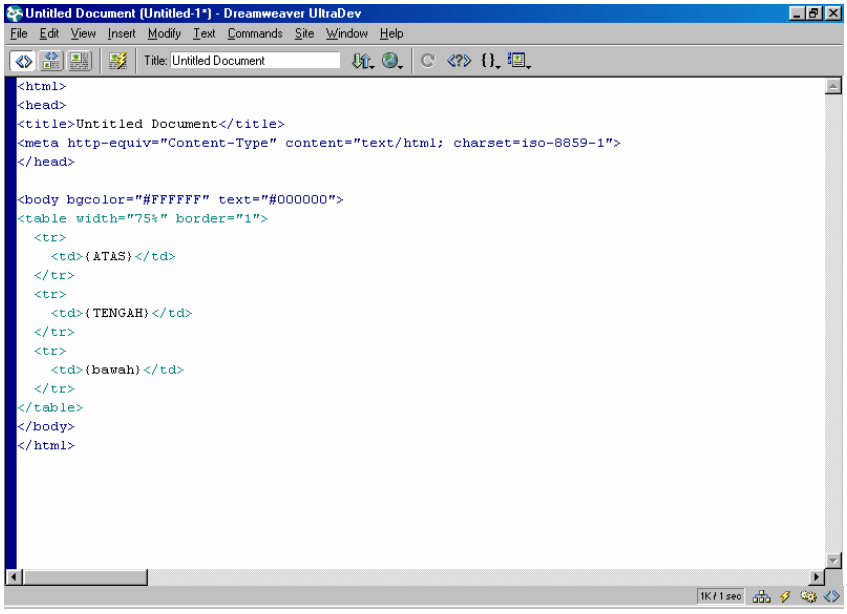
Tag dalam file template dimulai dengan tanda kurung '{' dan diakhiri dengan tanda kurung '}' .Sedangkan nama tag yang dituliskan diantara dua tanda kurung tersebut adalah suatu string biasa.

Untuk lebih jelasnya dapat dilihat pada gambar berikut :



Gambar 8.2 File template dan tag-tagnya (dilihat secara visual)

Kalau dilihat dalam format HTML adalah seperti berikut :



Gambar 8.3 File Template dilihat dalam format HTML

Gambar 8.2 dan 8.3 merupakan contoh file template misalnya diberi nama **template.tpl.php**. Dalam file template tersebut terdapat beberapa tag. Tag – tag tersebut nantinya akan menjadi indeks dari array associative. Untuk lebih jelasnya dapat dilihat dalam tabel berikut

Tabel 1. Tabel tag dalam file template dengan array

Tag dalam file template	Variabel array (var array = $a)
{ATAS}	$a[“ATAS”]
{TENGAH}	$a[“TENGAH”]
{bawah}	$a[“bawah”]

Ada hal penting yang harus diingat dalam penuliasan tag-tag yaitu penulisan nama tag bebas (boleh huruf kecil atau huruf besar) namun harus bersesuaian dengan index variable array.

b. Pembuatan Fungsi

Pembuatan fungsi untuk pemrograman template seperti halnya dalam pembuatan fungsi PHP biasa. Hanya saja dalam pembuatan fungsi kali ini

diperlukan pengembalian nilai keluaran fungsi dalam format *string*. Hasil keluaran fungsi (dalam format *string*) nanti akan dimasukkan dalam array assosiatif dengan indexnya adalah nama tag yang ada dalam file template.

Contoh fungsi yag digunakan dalam file template

```
<?
Function fungsiAtas() {
        $atas = date();

        return $atas;
}
Fungtion fungsiTengah($nama) {
        $tengah = “Hello Nama ku = “.$nama;

        return $tengah;

}

Fungtion fungsiBawah() {
        $bawah = “Adalah Mahasiswa Unwim”;

        return $bawah;
}
?>
```

c. Pengisian variabel array

Dalam pengisian variabel array ada beberapa hal yang harus diperhatikan yaitu

- Pengisian variabel array secara langsung
  Pengisian variabel array secara langsung dapat dilakukan sebagai berikut :

  $a[“ATAS”] = “hallo php”;
  akan menampilkan string hallo php dalam browser tepat pada posisi tag {USER} pada file template.

  $a[“ATAS”] = ‘<input type=”text” name=”jawab”>’;
  akan menampilkan text box dalam browser.

  Catatan :
  Penggunaan tanda ‘’ dan “” perlu diperhatikan sebagai berikut :
  - Tanda ‘’ digunakan jika diantara tanda tersebut ada tag html yang biasanya ada tanda “” dalam tag attributnya

Contoh :
$a[“ATAS”] = ‘<input type=”text” name=”jawab”>’;

bisa juga digunakan tanda “” tetapi ada penambahan tanda slash dalam tag attribut HTML, contoh nya :

$a[“ATAS”] = “<input type=\”text\” name=\”jawab\”>”;

- Tanda “” digunakan untuk string biasa seperti contoh hallo php

- Pengisian variabel array dari pengembalian suatu fungsi
  Pengisian variabel array dari pengembalian suatu fungsi dapat dilihat pada contoh berikut :

  $a[“ATAS”] = fungsiAtas();
  $a[“TENGAH”] = fungsiTengah(“Arie”);
  $a[“bawah”] = fungsiBawah();

- Penggabungan *string* (HTML) dengan PHP dalam template

Dalam pemrograman menggunakan template penggabungan beberapa string yang terpisah oleh perintah php dapat dilakukan dengan menggunakan tag **‘..’** . Hal ini dapat dilihat dalam contoh berikut :

```
$a[“ATAS”] = ‘Sekarang tanggal ’.date().’ Ada acara pelatihan’;
```

Selain cara tersebut penggabungan dapat pula dilakukan dengan menggunakan tag **‘;** hal ini biasanya digunakan dalam fungsi, contohnya :

```
Function fungsiAtas() {
    $atas = ‘Pencetakan bilangan dari 1 sampai 10’;

    $atas .= ‘<br>’;

    for ($c=1;$c<11;$c++) {
        $atas .= $c.’<br>’;
    }

    $atas .= ‘Pencetakan bilangan telah selesai’;

    return $atas;

}
```

Catatan lain :
Oleh karena pengembalian nilai fungsi adalah string maka tidak diperkenankan menggunakan perintah echo dan print dalam fungsi. Adanya perintah echo atau print dapat menyebabkan pencetakan *string* langsung ke browser tanpa melalui proses *parsing*.

d. *Parsing*

Dalam pemrograman template untuk mem-*parsing* file template digunakan fungsi Parse([file template],[array]). Pemanggilan fungsi *parsing* dilakukan sebagai berikut :

```
<?
    Parse (“template.tpl.php”,$a);
?>
```

dimana file template adalah template.tpl.php, dan variable array assosiatif adalah $a.

Untuk argumen file template termasuk didalamnya adalah *path* dari file template tersebut.

3. Contoh Kasus

a. Program 1

- Pembuatan File Template
  Buatlah file template seperti berikut :

```
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>

<body bgcolor="#FFFFFF" text="#000000">
<table width="75%" border="1">
  <tr>
    <td colspan="2">{TITLE}</td>
    <td width="38%">
      <div align="right">{TANGGAL}</div>
    </td>
  </tr>
  <tr>
    <td colspan="3">Selamat Datang {NAMA}</td>
  </tr>
  <tr>
    <td width="20%">{MENU}</td>
    <td colspan="2">{ISI}</td>
  </tr>
</table>
</body>
</html>
```

simpan dengan nama file template1.tpl.php

- Pembuatan Fungsi

```
function title() {
      $title = "Program Pelatihan PHP";

      return $title;
}

function tanggal() {
```

```
        $tanggal = date("j-n-Y");
        return $tanggal;
    }

    function nama($nama) {

        return $nama;
    }

    function menu() {
        $menu = '<a href="'.$PHP_SELF.'?v=nama1">nama1</a>'.'<br>';
        $menu .= '<a href="'.$PHP_SELF.'?v=nama2">nama2</a>';

        return $menu;
    }

    function isi($var_isi) {

        switch ($var_isi) {
           case "nama1" :
               $isi .= "anda telah mengklik nama pertama"."<br>";
               break;
           case "nama2" :
               $isi .= "anda telah mengklik nama kedua"."<br>";
               break;
           default :
               $isi .= "silakan klik menu di samping"."<br>";
               break;
        }

        return $isi;
    }
```

- Pengisian array

Misal variable array yang akan digunakan adalah $a, maka :

```
$a[“TITLE”] = title();
$a[“TANGGAL”] = tanggal();
$a[“NAMA”] = nama(“Arie”);
$a[“MENU”] = menu();
$a[“ISI”] = isi($v);
```

- Proses Parsing

Untuk mem-parsing file template maka digunakan fungsi Parse.

```
Parse (“template1.tpl.php”,$a);
```

Jangan lupa untuk memasukkan (include) file parsing.php

Secara lengkap fungsi, pengisian array dan parsing dimasukkan dalam file latih1.php :

```
<?
include ("parsing.php");

function title() {
    $title = "Program Pelatihan PHP";

    return $title;
```

```
}

function tanggal() {
	$tanggal = date("j-n-Y");
	return $tanggal;
}

function nama($nama) {

	return $nama;
}

function menu() {
	$menu = '<a href="'.$PHP_SELF.'?v=nama1">nama1</a>'.'<br>';
	$menu .= '<a href="'.$PHP_SELF.'?v=nama2">nama2</a>';

	return $menu;
}

function isi($var_isi) {

	switch ($var_isi) {
	   case "nama1" :
	        $isi .= "anda telah mengklik nama pertama"."<br>";
	        break;
	   case "nama2" :
	        $isi .= "anda telah mengklik nama kedua"."<br>";
	        break;
	   default :
	        $isi .= "silakan klik menu di samping"."<br>";
	        break;
	}

	return $isi;
}

$a["TITLE"] = title();
$a["TANGGAL"] = tanggal();
$a["NAMA"] = nama("Arie");
$a["MENU"] = menu();
$a["ISI"] = isi($v);

Parse ("template1.tpl.php",$a);

?>
```

b. Program 2

- Persiapan
  1. Buat database uji
  2. Buat table mahasiswa dengan fields :

Fields	Tipe	Panjang
nim	varchar	50
nama	varchar	100
tempat_lahir	varchar	100
tanggal_lahir	date	

Dengan primary key adalah nim.

Isikan data berikut :

nim	nama	tempat_lahir	tanggal_lahir
111	Arie	bandung	12 Mei 1990
222	Mio	bandung	13 Mei 1990
333	Nobita	tokyo	13 April 1990
444	Atik	semarang	14 April 1990

- Pembuatan File Template
  Buat file template berikut dan simpan dalam file template2.tpl.php

```
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>

<body bgcolor="#FFFFFF" text="#000000">
<table width="75%" border="0" bgcolor="#cccccc">
  <tr bgcolor="#000099">
    <td bgcolor="#000099"><font color="#FFFFFF">{TITLE}</font></td>
  </tr>
  <tr>
    <td>{ISI}</td>
  </tr>
  <tr>
    <td> </td>
  </tr>
  <tr>
    <td>{FOOTER}</td>
  </tr>
</table>
</body>
</html>
```

- Pembuatan Fungsi, Array dan *parsing*

Buat file php berikut dan simpan dalam file latih2.php

```
<?
include ("parsing.php");

function title() {
	$title = "Daftar Mahasiswa";

	return $title;
}

function isi() {
	// variabel-variabel
	// -- nama host
	$host = "192.168.0.248";
	// -- nama user
	$user = "root";
	// -- password
	$pass = "cyber";
	// -- nama database
	$namadb = "uji";

	// koneksi ke database MySQL
	$link = mysql_connect($host,$user,$pass) or die ("Tidak bisa koneksi");
```

```
        // pilih database
        mysql_select_db($namadb,$link);

        // query to mysql
        $sql = "select * from mahasiswa";
        $result = mysql_query($sql,$link);

        $isi .= '
        <table width="100%" border="1" cellpadding="0" cellspacing="0">
          <tr>
             <td width="21%" bgcolor="#AAAAAA"><b>NIM</b></td>
             <td width="27%" bgcolor="#AAAAAA"><b>NAMA</b></td>
             <td width="25%" bgcolor="#AAAAAA"><b>TEMPAT LAHIR</b></td>
             <td width="27%" bgcolor="#AAAAAA"><b>TANGGAL LAHIR</b></td>
          </tr>
        ';

       while ($row = mysql_fetch_array ($result)) {
        $isi .= '
          <tr>
             <td width="21%"> '.$row["nim"].'</td>
             <td width="27%"> '.$row["nama"].'</td>
             <td width="25%"> '.$row["tempat_lahir"].'</td>
             <td width="27%"> '.$row["tanggal_lahir"].'</td>
          </tr>
        ';
       }
        $isi .='
        </table> ';

        return $isi;
}

function footer() {
    $footer = "Dibuat oleh admin";

    return $footer;
}

$a["TITLE"] = title();
$a["ISI"] = isi();
$a["FOOTER"] = footer();

Parse ("template2.tpl.php",$a);

?>
```

Keterangan :
Variable-variabel dalam fungsi isi() dapat diganti disesuaikan dengan kondisi masing-masing (hostnya,user dan password) :

```
// variabel-variabel
        // -- nama host
        $host = "192.168.0.248";
        // -- nama user
        $user = "root";
        // -- password
        $pass = "cyber";
        // -- nama database
        $namadb = "uji";
```