BAN CHỈ ĐẠO CÔNG NGHỆ THÔNG TIN CỦA CƠ QUAN ĐẢNG

****

TÀI LIỆU THAM KHẢO

NGÔN NGỮ LẬP TRÌNH PHP

HÀ NỘI, 2003

PHẦN I : GIỚI THIỆU NGÔN NGỮ LẬP TRÌNH PHP

CHƯƠNG I : NGÔN NGỮ LẬP TRÌNH PHP

I. Giới thiệu PHP và môi trường lập trình web.

1. PHP là gì ?

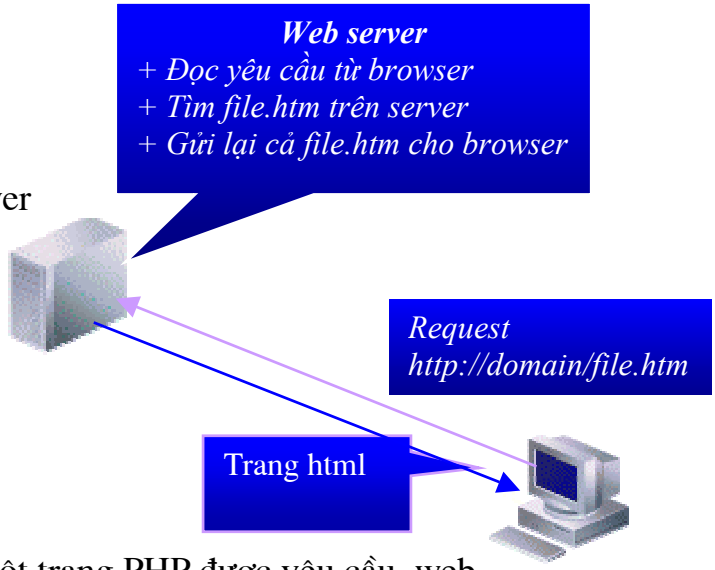
Cái tên PHP ban đầu được viết tắt bởi cụm từ Personal Home Page, và được phát triển từ năm 1994 bởi Rasmus Lerdorf. Lúc đầu chỉ là một bộ đặc tả Perl, được sử dụng để lưu dấu vết người dùng trên các trang web. Sau đó, Rasmus Lerdorf đã phát triển PHP như là một máy đặc tả (Scripting engine). Vào giữa năm 1997, PHP đã được phát triển nhanh chóng trong sự yêu thích của nhiều người. PHP đã không còn là một dự án cá nhân của Rasmus Lerdorf và đã trở thành một công nghệ web quan trọng. Zeev Suraski và Andi Gutmans đã hoàn thiện việc phân tích cú pháp cho ngôn ngữ để rồi tháng 6 năm 1998, PHP3 đã ra đời (phiên bản này có phần mở rộng là *.php3). Cho đến tận thời điểm đó, PHP chưa một lần được phát triển chính thức, một yêu cầu viết lại bộ đặc tả được đưa ra, ngay sau đó PHP4 ra đời (phiên bản này có phần mở rộng không phải là *.php4 mà là *.php). PHP4 nhanh hơn so với PHP3 rất nhiều. PHP bây giờ được gọi là PHP Hypertext PreProcesor.

2. Tại sao phải sử dụng PHP

Như chúng ta đã biết, có rất nhiều trang web được xây dựng bởi ngôn ngữ HTML (HyperText Markup Language). Đây chỉ là những trang web tĩnh, nghĩa là chúng chỉ chứa đựng một nội dung cụ thể với những dòng văn bản đơn thuần, hình ảnh ,và có thể được sự hỗ trợ bởi ngôn ngữ JavaScript, hoặc Java Apple. Những trang web như vậy người ta thường gọi là client-side. Tuy nhiên, Internet và Intranets đã được sử dụng cho các ứng dụng cần tới cơ sở dữ liệu. Các trang ứng dụng như vậy được gọi là trang web động, bởi vì nội dung của chúng luôn thay đổi tùy thuộc vào dữ liệu và người sử dụng. PHP là ngôn ngữ làm được những điều như vậy. Bằng cách chạy chương trình PHP trên máy chủ Web server, bạn có thể tạo ra các ứng dụng

có sự tương tác với cơ sở dữ liệu để tạo ra những trang web và đây được gọi là trang web động.

Chúng ta hãy xem xét cách hoạt động của trang web được viết bằng ngôn ngữ HTML và PHP như thế nào.



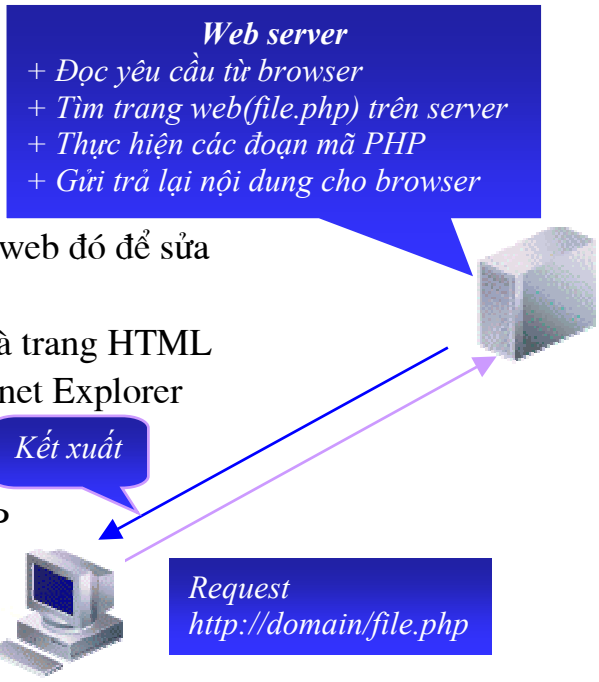
Với các trang HTML :

Khi có yêu cầu tới một trang web từ phía người sử dụng (browser). Web server thực hiệu ba bước sau :

+ Đọc yêu cầu từ phía browser,
+ Tìm trang web trên server.
+ Gửi trang web đó trở lại cho browser (nếu tìm thấy) qua mạng Internet hoặc Intranet .

Với các trang PHP :

Khác với các trang HTML, khi một trang PHP được yêu cầu, web server phân tích và thi hành các đoạn mã PHP để tạo ra trang HTML.



Điều đó được thể hiện bằng bốn bước sau :

+ Đọc yêu cầu tử phía browser.
+ Tìm trang web trên server.
+ Thực hiện các đoạn mã PHP trên trang web đó để sửa đổi nội dung của trang.
+ Gửi trở lại nội dung cho browser (đây là trang HTML có thể hiển thị được bởi trình duyệt Internet Explorer hoặc trình duyệt nào đó).

Tóm lại, sự khác nhau giữa HTML và PHP là HTML không được thực hiện trên máy chủ Web server còn các trang *.php viết bằng các đoạn mã PHP được thực hiện trên máy chủ Web server do đó nó linh động và mềm dẻo hơn .

3. Những điểm mạnh của PHP

-PHP thực hiện với tốc độ rất nhanh và hiệu quả .Một Server bình thường có thể đáp ứng được hàng triệu truy cập tới trong một ngày.

PHP hỗ trợ kết nối tới rất nhiều hệ CSDL khác nhau:

PostgreSQL,mSQL,Oracle, dbm, filePro ,Hyperware, informix,InterBase, Sybase, ... Ngoài ra còn hỗ trợ kết nối với ODBC thông qua đó có thể kết nối với nhiều ngôn ngữ khác mà ODBC hỗ trợ.

-PHP cung cấp một hệ thống thư viện phong phú : Do PHP ngay từ đầu được thiết kế nhằm mục đích xây dựng và phát triển các ứng dụng trên web nên PHP cung cấp rất nhiều hàm xây dựng sẵn giúp thực hiện các công việc rất dễ dàng : gửi, nhận mail ,làm việc với các cookie, và nhiều thứ khác nữa .

-PHP là một ngôn ngữ rất dễ dùng, dễ học và đơn giản hơn nhiều so với các ngôn ngữ khác như Perl, Java. Nếu bạn đã biết ngôn ngữ C thì mọi việc sẽ hoàn toàn thuận lợi .

-PHP có thể sử dụng được trên nhiều hệ điều hành, chúng ta có thể viết chúng trên Unix, Lunix và các phiên bản của Windows. Và có thể đem mã PHP này chạy trên các hệ điều hành khác mà không phải sửa đổi lại mã.

-PHP là ngôn ngữ mã nguồn mở.

II. Biến, hằng số và kiểu dữ liệu trong PHP.

1. Kiểu dữ liêu .

PHP có ba kiểu dữ liệu cơ bản : interger, double và string. Ngoài ra còn có các kiểu dữ liệu khác (nhưng không phải các kiểu dữ liệu cơ bản) như arrays (các kiểu dữ liệu mảng), objects (các kiểu dữ liệu đối tượng).

Interger là kiểu chiếm 4 byte bộ nhớ ,giá trị của nó trong khoảng –2 tỷ tới + 2 tỷ. Kiểu Double là kiểu số thực ,phạm vi biểu diễn $\pm (10^{-308} \div 10^{308})$. Kiểu string dùng để chứa các giá trị bao gồm các ký tự và con số .

Ví dụ : 2 // đây là kiểu interger

1.0 // đây là kiểu double

“2” // đây là kiểu string

"2 hours" // đây là một kiểu string khác

2. Hằng số

Hằng số là những giá trị không đổi. Chúng ta thường dùng hằng số để lưu các giá trị không đổi trong suốt chương trình như : nhiệt độ (0^0C), các giá trị thời gian chỉ sự chuyển giao giữa sáng ,chưa ,chiều ,tối ...

a. Khai báo hằng số :

Ta dùng hàm **define()** để khai báo hằng số :

define("COMPANY", "Phop's Bicycles");

define("YELLOW", "#FFFF00");

define("VERSION", 4);

define("NL", "
\n");

Trong ví dụ trên chúng ta đã dùng hàm define() để khai báo hằng số NL. Hằng số này là một thẻ ngắt dòng trong HTML.

Chúng ta sẽ sử dụng các hằng số trong PHP như sau :

echo ("Employment at ". COMPANY. NL);

Cách viết trên cũng giống như các viết sau:

echo ("Employment at Phop's Bicycles
\n");

Chú ý : hằng số phải ở ngoài hai dấu " và ". Trờng hợp sau là không có hiệu lực : echo ("Employment at COMPANY NL");. Khi thực hiện nó sẽ cho kết quả là : "Employment at COMPANY NL".

Hàm **defined()** : hàm này dùng để kiểm tra xem một hằng số nào đó đã được khai báo chưa.

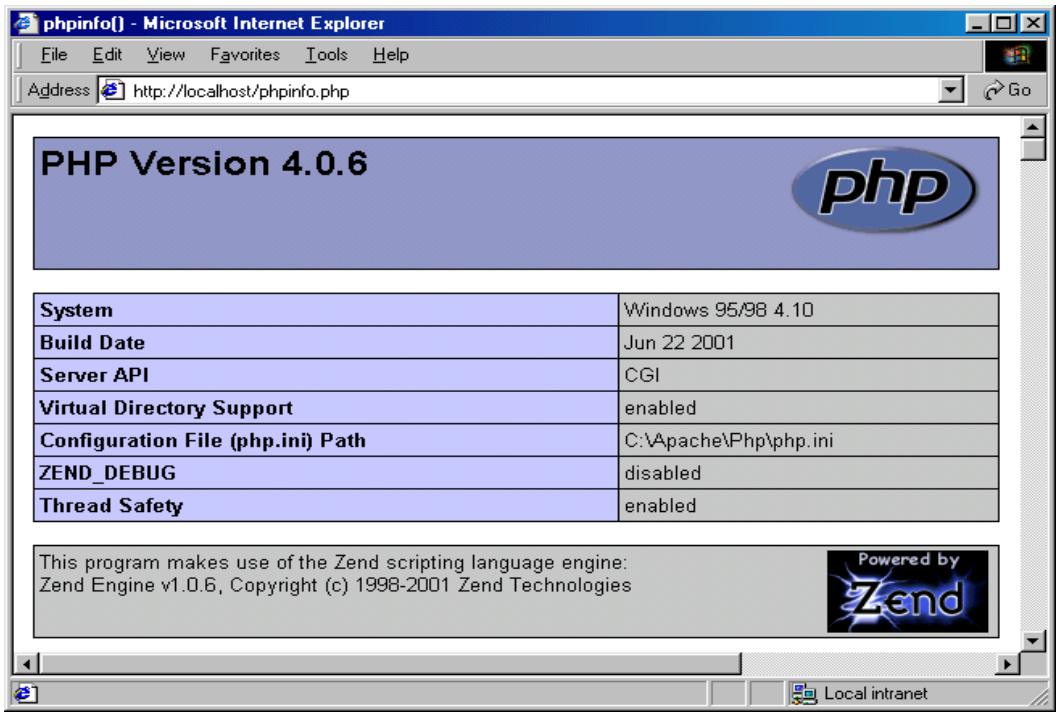
Ví dụ : if (defined ("YELLOW")) {

```
    echo ("<BODY BGCOLOR=". YELLOW. ">\n");
}
```

b. Các hằng số đã được định nghĩa trong PHP (Built-in Constants):

Để hỗ trợ cho người lập trình, PHP cung cấp sẵn các hằng số như : các biến môi trờng, các biến của Web server Apache ... Ngời lập trình có thể sử dụng hàm **phpinfo()** để xem các giá trị này.

```
<HTML>
<!—phpinfo() -->
<BODY>
        <?php  phpinfo();   ?>
</BOCY>
</HTML>
```



+ Hằng số nguyên : đây là những giá trị có kiểu integer. Ví dụ : 10

+ Hằng số thực: đây là những giá trị có kiểu double. Ví dụ : 10.00

+ Hắng ký tự : đây là một xâu ký tự đặt trong dấu ngoặc đơn hoặc kép.

Ví dụ : “Ngôn ngữ lập trình PHP”.

3. Biến và giá trị logic.

+ Cũng giống với C/C++, PHP không có khái niệm TRUE và FALSE. Các giá trị TRUE được hiểu là những giá trị bằng 1 và giá trị FALSE là những giá trị bằng 0 hoặc xâu rỗng .

+ Khi sử dụng biến chúng ta không cần khai báo kiểu .

Ví dụ : $a = 1; // $a là một biến kiểu integer.

$a = 1.2; // bây giờ $a là một biến kiểu double.

$a = "A" ; // bây giờ $a lại là một biến kiểu string.

+ Nếu như thực hiện phép toán giữa biến có kiểu số và kiểu string, PHP sẽ coi chuỗi là một dãy số như sau :

$str = "222B Baker Street";

Ta thấy biến $str có giá trị kiểu string, và nếu cộng số 3 với giá trị này thì :

$x = 3 + $str ; // $x = 225

khi đó biến $x nhận được giá trị 255 vì PHP đã cộng 3 với ba số đầu. Nhng nếu ta in giá trị của biến $str thì

echo ($str); // print : "222B Baker Street"

Chú ý rằng các phép toán giữa số và chuỗi chỉ đúng khi ký tự đầu của chuỗi là số .

+ Ta cũng có thể làm thay đổi kiểu giá trị của một biến bằng cách ép kiểu

$a = 11.2; // biến $a có kiểu double

$a = (int) $a; // bây giờ $a có kiểu integer ,giá trị là 11

$a = (double) $a; // bây giờ $a lại có kiểu double, giá trị là 11.0

$b = (string) $a ; // biến $b có kiểu string ,giá trị là "11"

Cũng phải biết rằng PHP tự động chuyển đổi kiểu rất tốt. Nếu thật sự cần thiết chúng ta mới phải dùng cách trên.

+ Các hàm làm việc với biến

gettype() : hàm này trả lại kiểu của một biến nào đó. Giá trị trả về có thể là : "integer"

"double"

"string"

"array"

"object"

"class"

"unknown type"

ví dụ :

```
if (gettype($user_input) == "integer")
{
        $age = $user_input;
}
```

settype() : hàm này ép kiểu cho một biến nào đó. Nếu thành công hàm trả về giá trị 1 (true) ,ngược lại là 0 (false).

ví du :

```
$a = 7.5;
settype($a, "integer");
if (settype($a, "array")){
        echo ("Conversion succeeded. ");
}else{
        echo ("Conversion error. ");
}
```

isset() và unset() : Hàm isset() kiểm tra một biến đã được gán giá trị hay chưa, hàm unset() sẽ giải phóng bộ nhớ cho một biến nào đó .

ví dụ :

```
$id = “323bb”;
if (isset($id)) {
        echo (“Dữ liệu đã được gán”);
}else{
        echo (“Dữ liệu chưa được gán”);
}
        unset($id);
        if(!isset($id)) {
                echo (“Dữ liệu đã được giải phóng”);
}
```

empty() : Cũng giống hàm isset(), hàm empty() sẽ trả về giá trị 1 (true) nếu một biến là rỗng và ngược lại 0 (false). Đối với biến có kiểu số giá trị bằng 0 được coi là rỗng, biến kiểu string được coi là rỗng nếu xâu là xâu rỗng.

ví dụ:

```
        echo empty($new) ;          // true
        $new = 1;
        echo empty($new);           // false
        $new = “”;
        echo empty($new);           // true
        $new = 0;
        echo empty($new);           // true
        $new = “So 323”;
        echo empty($new);           // false
        unset($new);
        echo empty($new);           // true
```

III. Các toán tử

+ Bảng các phép toán số học

Phép toán	ý nghĩa	Ví dụ	Giải thích
+	Phép cộng	7 + 2	Thực hiện phép cộng giữa 7 và 2 : 9
-	Phép trừ	7 – 2	Thực hiện phép trừ giữa 7 và 2 : 5
*	Phép nhân	7 * 2	Thực hiện phép nhân giữa 7 và 2 : 14
/	Phép chia	7 / 2	Thực hiện phép chia giữa 7 và 2 : 3.5
%	Chia d	7 % 2	Thực hiện phép chia d giữa 7 và 2 : 1

Ta có thể viết các phép toán ngắn gọn như bảng sau :

Khi viết	Tương đương với
$h += $i	$h = $h + $i
$h -= $i	$h = $h - $i
$h *= $i	$h = $h * $i
$h /= $i	$h = $h / $i
$h %= $i	$h = $h % $i

+ Phép gán :

ví dụ :

```
$x = 1;
$y = $x + 1;
$length = $area / $width;
```

+ Bảng các phép toán quan hệ

Phép toán	**ý nghĩa**	**Ví dụ**	**Giải thich**
==	So sánh bằng	$h == $i	Kiểm tra $h và $i có bằng nhau không
<	So sánh nhỏ hơn	$h < $i	Kiểm tra $h có nhỏ hơn $i không
>	So sánh lớn hơn	$h > $i	Kiểm tra $h có lớn hơn $i không
<=	Nhỏ hơn hoặc bằng	$h <= $i	Kiểm tra $h có nhở hơn hoặc bằng $i
>=	Lớn hơn hoặc bằng	$h => $i	Kiểm tra $h có lớn hơn hoặc bằng $i
!=	So sánh khác	$h != $i	Kiểm tra $h có khác $i không
<>	So sánh khác	$h <> $i	Kiểm tra $h có khác $i không

Các phép so sánh thường dùng kiểm tra điều kiện trong các câu lệnh điều khiển mà ta sẽ học ở bài sau .

+ Bảng các phép toán logic

Phép toán logic cùng với toán hạng tạo thành biểu thức logic. Biểu thức logic có thể có giá trị là 1 (true) hoặc 0 (false) .

Toán hạng a	**Toán hạng b**	**a && b**	**a || b**	**!a**	**!b**
1	1	1	1	0	0
1	0	0	1	0	1
0	1	0	1	1	0
0	0	0	0	1	1

+ Các phép toán với biến kiểu string .

Ta sử dụng dấu ". " để ghép hai biến kiểu string với nhau .

ví dụ : $first = "Phineas";

$last = "Phop";

$full = $first. " ". $last; // $full = "Phineas Phop" ;

echo ($full);

Ta có thể ghép hai xâu như sau:

echo ($last. "'s Bicycles"); //print : Phop's Bicycles

Để có thể chèn một biến vào trong hàng có kiểu string thì tên biến phải để trong dấu đóng mở ngoặc nhọn.

echo ("${last}'s Bicycles");

+ Các phép toán thao tác mức bit.

Các phép toán thao tác mức bit tác động lên từng bit của toán hạng .

Ký hiệu	ý nghĩa
&	AND bit
|	OR bit
^	XOR bit

Bảng các phép toán như sau :

&	Kết quả	|	Kết quả	^	Kết quả
1&1	1	1 | 1	1	1^1	0
1&0	0	1 | 0	1	1^0	1
0&1	0	0 | 1	1	0^1	1
0&0	0	0 | 0	0	0^0	0

PHP cũng hỗ trợ các phép dịch phải và dịch trái

>> : dịch phải

<< : dịch trái

Giả sử $a là một biến nguyên thì phép toán : $a >> n làm cho các bit trong $a bị dịch phải đi n vị trí. Tương tự ta có phép dịch trái .

ví dụ : 11 >> 2 = 2 ;

vì :

11 (1011)

>> 2

2 (0010)

+ Các phép toán tăng giảm :

- Phép tăng : phép tăng (toán tử tăng) tăng giá trị của toán hạng lên một đơn vị.

$a ++ : $a được sử dụng rồi mới tăng

++ $a : $a tăng rồi mới được sử dụng

- Phép giảm : tương tự như phép tăng, khác là giá trị bị giảm đi một đơn vị.

$a -- : $a được sử dụng rồi mới giảm

-- $a : $a giảm rồi mới được sử dụng

ví dụ :

```
$a = 10;       // $a bằng 10
$b = $a++ ;  // $a bằng 11 nhưng $b bằng 10
$a = 10;       // $a bằng 10
$b = -- $a ;   // $a bằng 9 và $b bằng 9
```

+ Phép toán điều kiện. ? :

Phép toán điều kiện cùng với toán hạng tạo nên biểu thức điều kiện. Ta ký hiệu e1 ,e2, e3 là ba toán hạng.

Biểu thức có dạng : e1 ? e2 : e3

Nếu e1 != 0 thì giá trị của biểu thức điều kiện là e2

Nếu e1 == 0 thì giá trị của biểu thức điều kiện là e3

ví dụ : tìm max

```
max = $a>$b ? a : b ;
```

+ Toán tử sizeof (đối tượng) :

Phép toán sizeof cho biết kích thước (tính bằng byte) ô nhớ mà đối tượng chiếm trong bộ nhớ. Đối tượng ở đây có kiểu là integer, double, string.

ví dụ : $a = 10;

```
echo sizeof($a);     //sẽ in ra màn hình là : 4
```

IV. CÁC CÂU LỆNH ĐIỀU KHIỂN

1. Lệnh if_else : đây là lệnh rẽ nhánh có điều kiện .

a. Dạng 1 :

```
if ( biểu thức ) câu lệnh;
```

Câu lệnh ở đây tương đương với một khối lệnh. Một khối lệnh được đặt trong dấu ngoặc kép.

ý nghĩa :

+ Nếu biểu thức khác không ,thì câu lệnh được thực hiện.

+ Nếu biểu thức bằng không, thì câu lệnh không được thực hiện

b. Dạng 2 :

```
if ( biểu thức )
        câu lệnh 1;
else
        câu lệnh 2;
```

ý nghĩa :

+ Nếu biểu thức khác không ,thì câu lệnh 1 được thực hiện.

+ Nếu biểu thức bằng không, thì câu lệnh 2 được thực hiện

Chú ý :

* Câu lệnh 1 ở dạng 2 là lệnh if_else

+ Nếu lượng else bằng lượng If thì else thuộc về If gần nhất theo từng cặp từ trong ra ngoài.

Ví dụ :

```
$a = 10;  $b = 10;
$c = 3;  $d = 3;
$e = 12;  $f = 8;
if($a == $b)
if($c == $d)
    if($e == $f)
        $max = $e;
    else
        $max = $f;
else
        $max = $d;
else
        $max = $b
echo $max ;                    //printf max = 8
```

+ Nếu lượng else ít hơn lượng If thì else thuộc về If ngần nhất theo từng cặp từ trong ra ngoài.

Ví dụ :

```
<?php
if ($a == $b)
if ($c == $d)
        $max = 0
else
        $max = $d;
?>
```

Tương đương với :

```
<?php
if ($a == $b)

{       if ( $c == $d)
                $max = 0;
}
else
        $max = $b;
?>
```

* Câu lệnh 2 của dạng 2 là elseif :

Bắt nguồn từ :

```
if ( biểu_thức1 )
  câu_lệnh 1;
else
  if ( biểu_thức 2 )
    câu_lệnh 2;
  else
    if ( biểu_thức 3 )
      câu_lệnh 3;
    ...
    else
      if ( biểu_thức i )
        câu_lệnh i;
      ...
      else
        câu_lệnh n;
```

Có thể viết lại như sau:

```
if ( biểu_thức1 )
  câu_lệnh 1;
elseif (biểu_thức 2)
    câu_lệnh 2;
elseif (biểu_thức 3)
    câu_lệnh 3;
    ...
elseif (biểu_thức i)
    câu_lệnh i;
    ...
else
    câu_lệnh n;
```

Câu lệnh elseif tạo ra lệnh rẽ nhánh có điều kiện trong đó thực hiện 1 trong n cách khác nhau.

- Nếu biểu_thức i khác không (i = 1,..n-1) thì thực hiện câu lệnh i .
- Nếu biểu_thức i bằng không (i = 1, ..n-1) thì câu lệnh thứ n được thực hiện.

2. Câu lệnh switch :

```
switch (biểu_thức n)
{
   case n1:
     câu lệnh 1;
     break;
   case n2:
     câu lệnh 2;
     break;
   ...
   case nn:
     câu lệnh nn;
  [default: câu lệnh]
}
```

Câu lệnh **switch** là câu lệnh rẽ nhánh có điều kiện trong đó thực hiện một trong n cái rẽ nhánh.

+Nếu biểu_thức n = ni (i = 1, .. n) thì câu lệnh sau *case ni* được thực hiện cho đến khi gặp lệnh break hoặc dấu ngoặc "}"

+Nếu biểu_thức n != mọi ni (i = 1, .. n) mà có nhánh *Default* thì các câu lệnh sau Default được thực hiện.

Ví dụ :

```
<?php
$val = 6;
$a = 5;
$b = 6;
$d = 7;
switch ($val)
{
        case $a :
                echo ("five");
                break;
        case $b :
                echo ("six");
                break;
        case $c :
                echo ("seven");
                break;
        default :
                echo ($val);
    }
?>
```

3. Câu lệnh FOR:

Dạng lệnh : **for** (biểu_thức 1; biểu_thức 2; biểu_thức 3)

câu lệnh ;

Lệnh **for** là lệnh tạo chu trình có điều kiện.

Bước 1 : tính biểu_thức 1

Bước 2 : tính biểu_thức 2 :

Nếu biểu_thức 2 khác 0 thì thực hiện câu lệnh và sang bước 3.

Nếu biểu_thức 2 bằng 0 thì kết thúc vòng for

Bước 3 : tính biểu_thức 3 và quay lại bước 2.

+ biểu_thức 1, biểu_thức 2, biểu_thức 3 là các thành phần. Mỗi thành phần có thể gồm nhiều biểu thức. Khi đó mỗi biểu thức được viết cách nhau một dấu phẩy (“,”).

+Các biểu thức được tính lần lượt từ trái qua phải

+Biểu thức trong biểu_thức 2 quyết định thực hiện thân của for.

Ví dụ :

```
<?php
        for($i = 0; $j = 4,$i < $j; $i++,$j--)
        {       echo (“i =”. $i. “ ,j = “. $j. “<br>”);
        }
?>
```

+Có thể vắng mặt bất kể thành phần nào. Nếu vắng mặt biểu_thức 2 thì câu lệnh luôn được thực hiện. Mặc dù vắng mặt vẫn phải có dấu chấm phẩy (“;”)

Ví dụ :

```
<?php
        for (  ;  ; ) {
        if (my_function() == “stop”) break;
        }
?>
```

+Nếu vắng biểu_thức 1 và biểu_thức 3 thì :

for (; biểu_thức 2 ;) cau_lenh ;

tương đương với :

while (biểu_thức 2) cau_lenh ;

4. Câu lệnh WHILE

while (biểu_thức)

câu lệnh ;

Lệnh **while** là một lệnh tạo chu trình có điều kiện. Điều kiện thực hiện được kiểm tra ở đầu chu trình.

Bước 1 : Tính biểu thức

Nếu biểu thức khác không, sang bước 2

Nếu biểu thức bằng không, kết thúc vòng while

Bước 2 : Thực hiện câu lệnh.

Quay lại bước 1.

Chú ý :

+ Biểu thức có thể bao gồm nhiều biểu thức. Khi đó các biểu thức được viết cách nhau một dấu phẩy ,và được tính lần lượt từ trái qua phải. Biểu thức cuối cùng quyết định thực hiện câu lệnh.

+ Không được phép vắng mặt biểu thức

+ Để tạo chu trình vô tận thì

```
while(1)
{       ...
        if (biểu_thức) break;
        ...
}
```

Ví dụ :

```
$i = 11;
while (--$i)
{       if (my_function($i) == “error”) {
                break;
        }
        ++ $number;
}
```

5. Lệnh DO ... WHILE

Dạng lệnh :

```
do {
    câu lệnh;
}while (biểu_thức);
```

Lệnh **do** ... **while** là lệnh tạo chu trình có điều kiện, trong đó điều kiện thực hiện chu trình được kiểm tra ở cuối chu trình.

Hoạt động :

Bước 1 : Thực hiện câu lệnh

Bước 2 : Tính biểu thức biểu_thức

+Nếu biểu thức biểu_thức khác không thì quay lại bước 1

+Nếu biểu thức biểu_thức bằng không thì kết thúc do ... while.

Ví dụ :

```
<?php
    echo (“<SELECT name=’num’>\n”);
    $i = 0;
    $total = 10;
    do {
        echo (“\t <OPTION value=$i>$i</OPTION>\n”);
}while(++ $i < $total);
echo (“</SELECT>\n”);
?>
```

6. Lệnh break

Là lệnh rẽ nhánh không điều kiện và thường dùng để ra khỏi thân của switch, while, do ... while, for .

Lệnh break chỉ cho phép thoát khỏi thân các lệnh bên trong nhất chứa nó.

7. Lệnh continue

Là lệnh rẽ nhánh không điều kiện .Lệnh thường dùng để bắt đầu lại một chu trình mới trong các lệnh for, while, do ... while mà không cần thực hiện hết toàn bộ thân của của lệnh tạo chu trình.

8. Khai báo tiền xử lý include và require .

Để sử dụng các đoạn mã ở bên ngoài, chúng ta có thể sử dụng khai báo tiền xử lý **include** và **require**. Cho phép chúng ta xây dựng các hàm các hằng số, và bất kỳ đoạn mã nào sau đó có thể chèn vào các đoạn script.

Require khác include là, nó có thể làm thay đổi nội dung của trang hiện tại khi biên dịch, các trang này dùng để khai báo các biến, các hằng số hay các đoạn mã đơn giản không có vòng lặp. Khi đó include cho phép thực hiện các câu lệnh phức tạp – có câu lệnh tạo chu trình. Nó chỉ sử dụng các hàm như những hàm ngoài của chương trình.

V. Hàm trong PHP

1. Quy tắc xây dựng hàm

function tên_hàm (danh sách đối số hình thức) {

Thân hàm .

}

+ Định nghĩa hàm không nhất thiết phải nằm ngoài thân mọi hàm, trong hàm có thể có hàm khác. Nhưng việc sử dụng một hàm không khác nhau giữa xây dựng hàm trong thân một hàm và ngoài mọi hàm .

+ Tên hàm tuỳ đặt và khác tên hàm chuẩn.

+ Hàm có thể có giá trị trả về hoặc không.

+ Các câu lệnh được quyền gọi bất kỳ hàm nào đã được khai báo và đã được định nghĩa.

+ Return : - Trả một giá trị về cho nơi gọi hàm

- Là nơi báo kết thúc hàm

2. Gọi hàm .

+ Hàm phải được xây dựng (khai báo) trước khi gọi .

+ Khi gọi hàm, nếu có giá trị trả về thường được đặt trong biểu thức .

3. Biến toàn cụ và biến cục bộ.

Thông thường PHP coi các biến được sử dụng trong thân của hàm là biến cục bộ. Nghĩa là biến trong thân hàm không làm thay đổi giá trị của biến ở ngoài hàm đó. Muốn hàm làm thay đổi giá trị của biến ngoài ta cần khai báo **global** trước biến đó trong thân hàm .

Ví dụ :

```
$position = “m”;
function change_pos()
{
        $position = “s”;
}
change_pos();
echo (“$position”);          //Prints “m”
```

Ta thấy biến $position giá trị không đổi sau khi gọi hàm change_pos().

```
$position = “m”;
function change_pos()
{       global $position;
        $position = “s”;
}
change_pos();
echo (“$position”);          //Prints “s”
```

Ta có thể viết như trên hoặc ta có thể viết :

```
$position = “m”;
function change_pos()
{       GOLOBALS[$position] = “s”;
}
change_pos();
echo (“$position”);          //Prints “s”
```

VI. Biến mảng trong PHP

1. Mảng một chiều

Mảng là một biến bao gồm nhiều phần tử có cùng tên nhưng khác nhau về chỉ số (các chỉ số này tăng dần từ 0 đến n). Với ngôn ngữ lập trình C, các phần tử của mảng có cùng kiểu dữ liệu ,nhưng với PHP thì mềm dẻo hơn. Các phần tử của mảng không nhất thiết phải cùng kiểu.

a. Khai báo mảng một chiều.

Ta có thể khai báo mảng bằng cách gán tên mảng với dấu đóng mở ngoặc vuông không có chỉ số. Chúng ta hãy xét ví dụ sau :

```
$countries[] = “cr”;
$countries[] = “de”;
$countries[] = “us”;
```

Ví dụ trên tạo ra một mảng gồm ba phần tử có chỉ số là 0, 1 và 2. Việc đó cũng tương tự như ta gán :

```
$countries[0] = “cr”;
$countries[1] = “de”;
$countries[2] = “us”;
```

Ngoài ra các chỉ số của mảng không nhất thiết phải tăng dần mà có thể được khai báo nh sau :

```
$countries[50] = “cr”;
```

$countries[20] = “de”;

$countries[10] = “us”;

echo (“$countries[20]”); // prints de

Khi đó để thêm một phần tử mới vào mảng chúng ta có thể viết

$countries[] = “uk”; // chỉ số sẽ là 51

Một phần tử mới được thêm vào với chỉ số là chỉ số lớn nhất của mảng cộng thêm một. Ngoài ra cũng có thể khai báo mảng một chiều bằng câu lệnh array

$countries = array (“cr”, “de”, “us”) ;

echo (“$countries[2]”); //prints “us”

Để chỉ số không bắt đầu từ không ta có thể khai báo lại như sau

$countries = array (1 => “cr”, “de”, “us”);

echo (“$countries[2]”); //prints “de”

Toán tử => có thể được sử dụng trước bất kỳ một phần tử nào trong mảng

$countries = array (“cr”, 7 => “de”, “us”);

và khi đó phần tử có giá trị “cr” có chỉ số là 0 còn phần tử có giá trị “de”, “us” lần lượt là 7 và 8. Khác với ngôn ngữ lập trình C, chỉ số của mảng một chiều không chỉ là các số nguyên mà còn là xâu ký tự. Ta có thể khai báo như sau :

$countries[“ca”] = “Canada”;

$countries[“cr”] = “Costa Rica”;

$countries[“de”] = “Germany”;

$countries[“uk”] = “United Kingdom”;

$countries[“us”] = “United States”;

echo (“$countries[“ca”]); // print Canada

Nếu dùng array thì sẽ là :

$countries = (“ca” => “Canada”,

```
"cr" => "Costa Rica",
"de" => "Germany",
"uk" => "United Kingdom",
"us" => "United States");
```

b. Làm việc với các phần tử của mảng

Có thể sử dụng vòng lặp for xét từng phần tử của mảng.

```
$countries = array (“cr”, “de”, “us”);
$num_elements = count($countries);
for ($i = 0 ; $i < $num_elements ; $i ++) {
        echo (“$countries[$i] <BR>\n”);
}
```

Ví dụ trên sẽ đúng khi chỉ số của các phần tử tăng dần đều. Nếu chỉ số các phần tử không theo một thứ tự nào cả ta có thể sử dụng hàm list() và each().

```
reset ($countries);
while (list ($key, $value ) = each ($countries)) {
        echo (“Chỉ số $key, giá trị $value”);
}
```

Để di chuyển con trỏ tới phần tử tiếp theo hoặc trước đó ta cũng có thể dùng hàm next() - hàm next() trả về là 1 (true) nếu phần tử tiếp theo không phải là phần tử cuối cùng, ngược lại là 0 (false) và prev() - hàm prev() cho giá trị trả về là 1 (true) nếu phần tử tiếp theo không phải là phần tử đầu tiên, ngược lại là 0 (false). Hàm key () sẽ cho biết chỉ số và hàm current() sẽ cho biết giá trị của phần tử đó .

```
$arr = array (3, 4, 5, 6, 7);
do {
        $k = key ($arr);
        $val = current ($arr);
        echo (“Phần tử $k = $val”);
}while
```

2. Mảng hai chiều

Chúng ta có thể khai báo mảng hai chiều như sau :

```
$countries = array (“Europs” => array (“de”, “uk”),
                    “North America” => array (“ca”, “cr”, “us”));
echo ($countries[“Europs”][1]);              // print “uk”
echo ($countries[“North America”][2]);       // print “us”
```

Ta có cấu trúc của mảng trên như sau :

$countries[“Europs”]		$countries[“North America”]		
[0]	[1]	[0]	[1]	[2]
“de”	“uk”	“ca”	“cr”	“us”

Cũng giống như mảng một chiều ta sẽ dùng vòng lặp như for, while, do ... while để duyệt qua các phần tử của mảng.

Ví dụ :

```
$countries = array (“Europs” => array (“de”, “uk”),
                    “North America” => array (“ca”, “cr”, “us”));
while (list ($key1) = each ($countries)) {
        echo (“$key1 : <BR>\n”);
        while (list ($key2, $val) = each($countries[“$key1”])) {
                echo (“ - $val <BR>\n”)
}
}
```

Khi chạy chương trình sẽ in ra màn hình là :

Europ :

- de
- uk

North America :

- ca
- cr
- us

3. Các hàm sắp xếp mảng .

PHP cung cấp cho chúng ta các hàm để sắp xếp mảng. Với mỗi loại mảng sẽ có một hàm tương ứng. Với mảng có chỉ số là kiểu nguyên chúng ta có hàm sort() để sắp xếp tăng dần các phần tử của mảng, hàm rsort() sẽ sắp xếp các phần tử của mảng giảm dần .

```
$countries = array (“us”, “uk”, “ca”, “cr”, “de”);
sort ($countries);
while (list ($key, $val) = each ($countries)) {
        echo (“Element $key equals $val <BR>\n”);
}
```

Khi chạy chương trình sẽ là :

Element 0 equals ca

Element 1 equals cr

Element 2 equals de

Element 3 equals uk

Element 4 equals us

Với mảng chỉ số có kiểu string thì dùng hàm asort (), arsort () để sắp xếp mảng theo chiều tăng dần và giảm dần. Nếu bạn dùng các hàm sort() và rsort() thì các chỉ số có kiểu string sẽ chuyển thành các chỉ số có kiểu nguyên.

```
 $countries = array(“us” => “United States”,
                    “uk” => “United Kingdom”,
                    “ca” => “Canada”,
                    “cr” => “Costa Rica”,
                    “de” => “Germany”);
asort ($countries);
while (list($key, $val) = each($countries)) {
        echo “Chi so $key bang $val <BR>\n”;
```

}

Kết quả khi chạy chương trình :

Chi so ca bang Canada

Chi so cr bang Costa Rica

Chi so de bang Germany

Chi so uk bang United Kingdom

Chi so us bang United States

Nhưng nếu thay dòng asort ($countries); bằng sort ($countries); kết quả sẽ là

Chi so 0 bang Canada

Chi so 1 bang Costa Rica

Chi so 2 bang Germany

Chi so 3 bang United Kingdom

Chi so 4 bang United States

Để sắp xếp mảng tăng dần hay giảm dần theo chỉ số chúng ta có hàm ksort() – tăng dần và krsort() – giảm dần .

```
$countries = array("e" => "United States",
                   "d" => "United Kingdom",
                   "c" => "Canada",
                   "b" => "Costa Rica",
                   "a" => "Germany");
ksort ($countries);
while (list($key, $val) = each($countries)) {
        echo "Chi so $key bang $val <BR>\n";
}
```

Kết quả :

Chi so a bang Germany

Chi so b bang Costa Rica

Chi so c bang Canada

Chi so d bang United Kingdom

Chi so e bang United States

VII. Lập trình hướng đối tượng (OOP)

1. Định nghĩa lớp .

Chúng ta có thể định nghĩa lớp bằng toán tử class, và trong mỗi một lớp chúng ta sẽ xây dụng các phương thức và thuộc tính cho lớp đó .

Giả sử muốn định nghĩa lớp "Sesssion" chúng ta làm như sau :

```
<?php
class Session
{
        // Định nghĩa các thuộc tính và phương thức
        // Xây dựng các phương thức cho lớp
}       // Kết thúc một lớp
?>
```

Để định nghĩa thuộc tính cho lớp chúng ta đặt từ khoá **var** trước thuộc tính đó.

```
<?
        class Session
        {
                var $sqlhost = "localhost";
                var $sqluser = "root";
                var $sqlpass = "password";
                var $sqldb = "session";
```

```
        var $linkid;
        var $seshid;
        var $sessdata;
        var $userid;
        var $error_no;
        var $err;
        ...
        // Định nghĩa các phương thức ở đây
    }   //Kết thúc định nghĩa lớp
?>
```

Tiếp theo chúng ta sẽ định nghĩa các phương thức cho lớp trên. Phương thức cần xây dựng đầu tiên là Session() sau đó đến các phương thức khác.

```
<?
// Định nghĩa lớp Session
class Session
{
    // Định nghĩa các thuộc tính
    .. .
    // Định nghĩa các phương thức
function Session($localSeshID, $localUserID=0)
{
$this->seshid = $localSeshID;
$this->userid = $localUserID;
// Kết nối tới MySQL
$this->linkid=mysql_connect($this->sqlhost, $this->sqluser, $this->sqlpass);
```

```
if (!$this->linkid)
    {
      $this->err=mysql_error();
      $this->error_no=102;
    }
}
...     // Định nghĩa các phương thức khác
}
?>
```

2. Sử dụng lớp đã được định nghĩa.

Để khai báo một đối tượng có kiểu thuộc lớp đã xây dựng ta dùng toán tử **new** như sau :

```
<?php
      require ("classes/sessions.php");     //include class
      $mysesh = new Session($seshid);
if ($mysesh->error_no)
      {
            //Thông báo có lỗi
      }
      $user = $mysesh->sessdata[userid];
?>
<HTML>
<HEAD>
<TITLE>Welcome to my website, <? echo $user ?></TITLE>
</HEAD>
<BODY>
<a href="nextpage.php?seshid=<? print $user ?>">Next Page </a>
```

```
<FORM ACTION=”nextpage.php” METHOD=”POST”>
<input type=”hidden” name=”seshid” value=”<? print $seshid ?>”>
</FORM>
</BODY>
</HTML>
```

VIII. Tìm hiểu thêm về HTML

Phần một chúng ta đã tìm hiểu sơ lược về ngôn ngữ HTML. Trong mục này chúng ta sẽ trình bày thêm về nó với trọng tâm là FORM của HTML. Lý do là vì khi muốn giao tiếp giữa HTML và PHP script thì không thể không biết đến FORM. FORM là công cụ phổ biến nhất để chuyển dữ liệu từ HTML đến PHP script (client lên server).
Thông thường, khi viết một script, người ta thiết kế một FORM sao cho nó cung cấp đầy đủ những thông tin mà script cần để xử lý.

1. Cấu trúc của một FORM

Một FORM được mở đầu bằng thẻ <FORM ... > và kết thúc bằng thẻ </FORM>.

Trong FORM có :

- Thuộc tính : FORM thường có ba thuộc tính, ba thuộc tính này nằm trong thẻ FORM mở đầu của một FORM : action, method, enctype .

- Các thẻ được dùng trong FORM : input, select, textarea, hn, p, hr, dir, dl, menu, ol, ul, address, blockquote, [isindex]. pre .

- FORM có ba thành phần chính : textarea, select, input .

- FORM có thể được dùng trong các thẻ : blockquote,body,dd, li.

2. Các thuộc tính

Khi tạo một FORM ,công việc đầu tiên là xác định thuộc tính của nó.

- Thuộc tính action : Action=”URL” chỉ đến script mà FORM sử dụng. Ngoài ra nó cũng có thể là mailto url, khi đó nội dung của FORM được mail đến địa chỉ trong url .

- Thuộc tính method : Method có thể được gán bằng “GET” hoặc “POST” ,nó chỉ ra phương thức Post hay Get sẽ được sử dụng trong FORM, giá trị ngầm định là Get.

- Thuộc tính enctype : Enctype=”Mime_type” :chỉ ra loại dữ liệu sẽ gửi đi. Giá trị ngầm định là application/x-www-form-urlencode.

Trong một trang chúng ta có thể sử dụng nhiều FORM nhng không được lồng chúng vào nhau.

2. Các thành phần cơ bản của FORM.

Một FORM ,thông thường có ba thành phần chính :

+Textarea

+Select

+Input

a - Textarea

Được bắt đầu bằng thẻ <Texterea> và kết thúc bằng thẻ </Texterea>. Bên trong chứa các ký tự, thẻ này được dùng lồng vào trong FORM.

Các thuộc tính của Texterea .

- Name : Định nghĩa tên của thành phần, thuộc tính này luôn luôn phải có.

- Rows : Cho biết số hàng của hộp văn bản.

- Cols : Cho biết số cột của văn bản.

Wrap : Chỉ ra cách xử lý word_wraping (căn chiều dài của dòng văn bản theo kích thước của hộp văn bản) trong hộp thoại. Nếu wrap=”off”, chức năng word_wraping bị cấm. Nếu wrap=”vitual”, chức năng này được bật lên, khi gõ văn bản vào, ta thấy con trỏ tự động xuống hàng mỗi khi nó chạy đến biên của hộp văn bản, ký tự newline (OA hex) tự động được thêm vào nhưng không được gửi đi cùng FORM. Nếu wrap=”physical”, chức năng này cũng được bật và hoạt động như trên nhưng ký tự newline được gửi đi cùng với FORM.

Văn bản nằm giữa hai thẻ <Texterea> và </Texterea> sẽ được thể hiện như đoạn văn bản ngầm định trong vùng dữ liệu.

Ví dụ:

<Texterea name=”van_ban” rows=”4” cols=”40”>

.. .

</ Texterea >

b. Select

Được bắt đầu bằng thẻ <Select> và kết thúc bằng thẻ </ Select >, cặp thẻ này có thể lồng trong FORM hay bất kỳ thành phần nào của FORM ngoại trừ Texterea và Select .

Select có các thuộc tính sau :

- Name : Tên của thành phần.

- Size : Cho biết số thành phần sẽ hiển thị, giá trị ngầm định là 1, do đó danh sách lựa chọn thường được thể hiện dới dạng pop-up menu. Thuộc tính này có thể không có.

- Multiple : Nếu thuộc tính này được thiết lập, nhiều lựa chọn sẽ được chọn cùng một lúc, ngược lại chỉ được chọn một item.

Khi sử dụng Select, chúng ta có thể dùng thêm Option

<Option>. .. .. .. </Option>

Bên trong chứa các ký tự, có thể được sử dụng lồng vào Select .

Các thuộc tính Option :

- Disable : Đánh một lựa chọn bị cấm. Khi hiển thị, chọn lựa này sẽ bị che mờ.

- Selected : Đánh dấu chọn lựa này đã được chọn, nếu thuộc tính Multiple được bật trong Select, bạn có thể đánh dấu Selected nhiều chọn lựa cùng lúc. Nó dùng để đánh dấu các lựa chọn ngầm định.

- Value : Chỉ ra giá trị được gán cho lựa chọn, nếu không có thì nội dung của thư mục option sẽ được gửi đi thay cho value.

c. Input

Mở đầu bằng thẻ < Input >, thẻ này có thể dùng trong bất cứ thành phần nào khác của FORM ngoại trừ Texterea và Select.

Các thuộc tính :

- Align : Có thể là một trong ba giá trị top, middle, bottom dùng để căn lề ảnh với các văn bản xung quanh, thuộc tính này chỉ có ý nghĩa với Type="image" .

- Name : gắn tên biến cho dữ liệu của thành phần này. Giá trị của thuộc tính do user lựa chọn .

- Type : định ra một trong những giá trị sau checkbox, hidden, image, password, radio, reset, submit, text, file, bottom .

- Checked : chỉ ra một nút radio, hay một checkbox cơ được chọn hay không.

- Maxlength : chỉ ra chiều dài tối đa mà hộp thoại văn bản có thể chứa, thuộc tính này chỉ có ý nghĩa với input có type="text" hay type="password" .

- Size : chỉ ra kích thước thực sự của hộp thoại văn bản.

IX. Tóm lược

Như vậy, chúng ta đã khảo sát hầu hết các chức năng cơ bản của PHP, từ các kiểu dữ liệu, khai báo biến, hàng, mảng cho tới cách thực hiện chương trình dạng Máy khách/Máy chủ (Client/Server), và nhúng các đoạn mã PHP vào các trang HTML .v.v.

Như trên chúng ta đã trình bày, cơ sở dữ liệu cho trang Web là thành phần không thể thiếu, nó đóng vai trò quyết định cho chương trình. Ví lý do đó, phần tiếp theo chúng ta sẽ khảo sát ngôn ngữ SQL và các hàm API của PHP để thực hiện các lệnh truy vấn trên hệ quản trị cơ sở dữ liệu MySQL.

X. Các hàm API trong PHP

1. Giới thiệu về MySQL .

PHP hỗ trợ một số lượng lớn các hàm làm việc với cơ sở dữ liệu nh Oracle, Sybase, PostgreSQL, MySQL. .. Thông qua chuẩn ODBC (Open Database Connectivity), bằng cách sử dụng các hàm API (Application

Programming Interface) mà PHP có thể làm việc được với nhiều hệ quản trị cơ sở dữ liệu như vậy. Nếu hệ quản trị cơ sở dữ liệu không hỗ trợ ODBC ,và hơn nữa ODBC có đặc điểm chỉ hỗ trợ ở dạng chuẩn thì PHP có thể làm việc với ODBC ở tầng trên. Nếu không muốn sử dụng ODBC ta có thể sử dụng các hàm API .

Trong phần này chúng ta chỉ sử dụng các hàm API để làm việc với hệ quản trị cơ sở dữ liệu MySQL .

2. Các hàm cơ bản làm việc với cơ sở dữ liệu MySQL.

a) Các hàm kết nối đến MySQL Server

PHP cung cấp hai hàm để kết nối với cơ sở dữ liệu MySQL : mysql_connect và mysql_pconnect .

+ mysql_connect () : hàm này sẽ tạo ra một liên kết tới máy chủ MySQL .

Cú pháp :

int mysql_connect (string [hostname [:port] [:/path_to_socket]], string [username], string [password]);

Trong đó :

- hostname : Tên máy chủ cơ sở dữ liệu, nơi trang web sẽ chứa cơ sở dữ liệu. Giá trị ngầm định là "localhost'"

- :port : Địa chỉ cổng, nơi bộ máy cơ sở dữ liệu lắng nghe yêu cầu. Giá trị ngầm định là ":3306" .

- :/path_to_socket : Cũng giống như :port nhưng chỉ cho hệ điều hành UNIX. Giá trị ngầm định là ":/tmp/mysql.sock" .

- username : Tên của người sử dụng được phép kết nối vào bộ máy cơ sở dữ liệu.

- password : Mật khẩu của người sử dụng để kết nối vào bộ máy cơ sở dữ liệu.

Hàm này trả về mã số nhận dạng nếu kết nối thành công, giá trị 0 (false) nếu việc kết nối có lỗi. Mã số nhận dạng này sẽ được sử dụng cho tất cả các yêu cầu tới bộ máy cơ sở dữ liệu sau này.

Sự kết nối này sẽ đóng lại khi gọi hàm mysql_close() hoặc kết thúc đoạn PHP script.

+ mysql_pconnect() : Hàm này tạo ra một liên kết bền vững với máy chủ MySQL.

Cú pháp :

int mysql_pconnect (string [hostname [:port] [:/path_to_socket]], string [username], string [password]);

Tham số và giá trị trả về của hàm này cũng giống hàm mysql_connect(). Sự khác biết giữa hai hàm này là liên kết tới máy chủ MySQL không bị đóng lại kể cả khi kết thúc kịch bản (script) PHP hay gọi hàm mysql_close(). Mục đích của hàm này là luôn luôn duy trì liên kết tới máy chủ MySQL do luôn có sự yêu cầu tới máy chủ, tránh cho máy chủ phải tìm kiếm mã số nhận dạng mới từ đó giảm thời gian truy cập .

Chú ý : hàm này chỉ thực hiện được khi PHP được định cấu hình như là một module của Web server .

+ mysql_close() : Hàm này huỷ bỏ sự kết nối tới máy chủ MySQL .

Cú pháp :

int mysql_close(int [link_identifier]);

Tham số link_identifier là mã số nhận dạng tạo ra bởi hàm mysql_connect(). Hàm trả về là True nếu thành công, ngược lại là False .

b) Các hàm thao tác trên CSDL

+ mysql_create_db() : Hàm tạo cơ sở dữ liệu

Cú pháp :

int mysql_create_db(string name, int *[link_identifier]*) ;

Trong đó :

- string name : Tên của cơ sở dữ liệu cần tạo.

- int link_identifier : Mã số nhận dạng được cấp bởi hàm mysql_connect() .

Chúng ta hoàn toàn có thể gửi câu lệnh SQL để tạo cơ sở dữ liệu thông qua hàm mysql_query() .

+ mysql_drop_db() : Hàm xoá cơ sở dữ liệu

Cú pháp :

int mysql_drop_db(string name, int *[link_identifier]*);

Trong đó :

- string name : Tên của cơ sở dữ liệu cần xoá .

- int link_identifier : Mã số nhận dạng được cấp bởi hàm mysql_connect() .

Chúng ta hoàn toàn có thể gửi câu lệnh SQL để xoá cơ sở dữ liệu thông qua hàm mysql_query().

+ mysql_select_db() : Hàm cho cơ sở dữ liệu hoạt động .

Cú pháp :

int mysql_select_db(string database_name, int *[link_identifier]*);

Trong đó:

- database_name : Tên của cơ sở dữ liệu mà sau này các hàm API khác của PHP sẽ thực hiện trên đó.

- int link_identifier : Mã nhận dạng được cấp bởi hàm mysql_connect().

Câu lệnh này sẽ gắn tên cơ sở dữ liệu với mã nhận dạng, sau này khi làm việc với link_identifier sẽ bao gồm cả cơ sở dữ liệu được chọn .

c) Các hàm thao tác trên dữ liệu

+ mysql_query() : Hàm gửi câu lệnh SQL tới máy chủ MySQL .

Cú pháp :

int mysql_query(string query, [int *link_identifier]*) ;

Trong đó :

- string query : Câu lệnh SQL cần gửi tới máy chủ MySQL .

- int link_identifier : Mã số nhận dạng, nó phải được thực hiện trong hàm mysql_select_db() trước đó .

+ mysql_db_query() : Hàm gửi câu lệnh SQL tới máy chủ MySQL .

Cú pháp :

int mysql_db_query(string database, string query, int *[link_identifier]*);

Trong đó :

- string database : Tên cơ sở dữ liệu câu lệnh SQL sẽ thực hiện trên đó.

- string query : Câu lệnh SQL cần thực hiện .

- link_identifier : Mã số nhận dạng được cấp bởi hàm mysql_connect()

Hàm này chỉ rõ câu lệnh được thực hiện trên cơ sở dữ liệu nào nên trước đó không cần thực hiện hàm mysql_select_db();

+mysql_insert_id() :

Hàm lấy giá trị được sinh ra từ câu truy vấn INSERT trớc

Cú pháp :

int mysql_insert_id(*[link_identifier]*) ;

trong đó:

int link_identifier : Mã số nhận dạng được cấp bởi hàm mysql_connect() .

Hàm này trả về giá trị id được sinh ra trong cột AUTO_INCREMENT bởi câu truy vấn trước đó. Điều này chỉ có tác dụng trên *link_identifier* được chỉ ra trong hàm, nếu gọi hàm trên mà không chỉ định tham số *link_identifier* thì liên kết được mở cuối cùng sẽ được chỉ định.

Hàm mysql_insert_id() trả về giá trị 0 nếu câu truy vấn trước đó không sinh ra một giá trị AUTO_INCREMENT. Nếu ta muốn giữ lại giá trị cho lần sau, thì phải gọi hàm này ngay sau câu truy vấn sinh ra giá trị .

+ mysql_fetch_row() :

Hàm trả về một mảng là giá trị của một bảng ghi hiện tại với chỉ số là số thứ tự của các trường (chỉ số bắt đầu từ 0). Sau đó hàm sẽ trỏ tới bảng ghi tiếp theo cho tới khi gặp bảng ghi cuối cùng hàm trả về giá trị false. Để truy xuất tới các giá trị của cột ta viết : tên_mảng[số thứ tự]

Cú pháp :

array mysql_fetch_row(int *result_identifier*);

Trong đó : result_identifier là mã số trả về của hàm mysql_query() hoặc mysql_db_query() .

Ví dụ :

```
<?php
    $mysql = "select id, name from ds_thanhvien"; // cau lenh SQL
    $link = mysql_connect($host, $user, $password); //lay ma
    mysql_select_db($database_name, $link);
    $result = mysql_query($mysql ,$link);
    while ($row = mysql_fetch_row($result))
      {
          echo $row[0] ;
          echo $row[1];
      }
?>
```

+ mysql_fetch_array() :

Hàm trả về một mảng là giá trị của một bảng ghi hiện tại, sau đó hàm sẽ trỏ tới bảng ghi tiếp theo cho tới khi gặp bảng ghi cuối cùng hàm trả về giá trị false.

Cú pháp :

array mysql_fetch_array(int *result_identifier [,* int result_type]);

Trong đó : result_identifier là mã số trả về của hàm mysql_query() hoặc mysql_db_query() .

Để truy xuất đến các thành phần của cột :

tên_biến_mảng["tên_trờng"];

result_type là một hằng số có thể nhận các giá trị sau:

-MYSQL_NUM : chỉ trả lại một mảng chứa các chỉ số là số (giống như hàm mysql_fetch_row())

-MYSQL_ASSOC: chỉ trả lại một mảng liên kết

-MYSQL_BOTH : trả lại mảng chứa đựng các chỉ số gồm cả các con số và chỉ số liên kết .

Hàm này là sự mở rộng của hàm mysql_fetch_row(). Nó cho phép truy cập trường dữ liệu của mảng kết quả không chỉ thông qua các chỉ số là các số mà chúng có thể là tên của các trường dữ liệu. Điều này làm cho việc lập trình đơn giản và chính xác hơn.

Ví dụ:

```
<?php
        $mysql = "select id, name from ds_thanhvien";
        $link = mysql_connect($host, $user, $password);
        $result = mysql_db_query("php", $mysql);
        while ($row = mysql_fetch_array($result)) {
                echo "user_id: ". $row["id"] ."<BR>\n";
                echo "user_id: ". $row[0] ."<BR>\n";
                echo "user_name: ". $row["name"] ."<BR>\n";
echo "user_name: ". $row[1] ."<BR>\n";
        }
        mysql_free_result ($result);
  ?>
```

+ mysql_fetch_object() : Hàm trả về một đối tượng là giá trị của một bảng ghi hiện thời. Sau đó hàm sẽ trỏ tới bảng ghi tiếp theo cho tới khi gặp bảng ghi cuối cùng hàm trả về giá trị false. Để truy xuất tới các giá trị của cột ta viết tên_object->tên_cột .

Cú pháp :

object mysql_fetch_object(int result_identifier);

Trong đó : result_identifier là mã số trả về của hàm mysql_query() hoặc mysql_db_query() .

Ví dụ :

```
<?php
  $mysql = “select id, name from ds_thanhvien”;
  $link = mysql_connect($host, $user, $password);
  $result = mysql_db_query(“php”, $mysql);
  while ($row = mysql_fetch_object($result)) {
        echo $row->id ;
        echo $row->name;
    }
?>
```

+mysql_fetch_assoc() :lấy về một dòng kết quả như là một mảng liên kết .

cú pháp: array mysql_fetch_assoc(int result_identifier)

Trong đó : result_identifier là mã số trả về của hàm mysql_query() hoặc mysql_db_query() .

Hàm trả về một mảng tương ứng với một bản ghi được lấy về và trả lại FALSE nếu không có bản ghi nào. Hàm này tương đương với hàm

array mysql_fetch_array() với tham số result_type là : MYSQL_ASSOC

ví dụ :

```
<?php
  $mysql = “select id, name from ds_thanhvien”;
  $link = mysql_connect($host, $user, $password);
  $result = mysql_db_query(“php”, $mysql);
  while ($row = mysql_fetch_assoc($result)) {
        echo $row[“id”];
        echo $row[“name”];
    }
?>
```

+mysql_data_seek()

Di chuyển con trỏ bên trong "tập kết quả" (có được sau khi câu truy vấn SELECT được thực hiện)

Cú pháp: bool mysql_data_seek(int result_identifier, int row_number);

Trong đó : result_identifier là mã số trả về của hàm mysql_query(), mysql_db_query(), mysql_list_tables(), mysql_list_dbs() .

row_number là chỉ số của bản ghi mà cần đặt con trỏ vào .

Hàm trả về true nếu thành công, false nếu lỗi .

Hàm này sẽ di chuyển con trỏ bên trong "tập kết quả" (được chỉ rõ bởi tham đối *result_identifier*) đến dòng có mã bằng tham đối *row_number*.

Các dòng trong tập kết quả được bắt đầu từ 0

Ví dụ:

```
<?php
   $link = mysql_pconnect ($host, $user, $password)
      or die ("Could not connect");

   $query = "SELECT last_name, first_name FROM friends";
   $result = mysql_db_query ("php",$query)
      or die ("Query failed");

   # fetch rows in reverse order

   for ($i = mysql_num_rows ($result) - 1; $i >=0; $i--) {
      if (! Mysql_data_seek ($result, $i)) {
         printf ("Cannot seek to row %d\n", $i);
         continue;
      }

      if(!($row = mysql_fetch_object ($result)))
         continue;

      printf ("%s %s<BR>\n", $row->last_name, $row->first_name);
   }

   mysql_free_result ($result);
?>
```

+ mysql_num_rows() :trả lại số dòng trong result_identifier (nơi chứa kết quả của câu lệnh SQL đã được thực hiện)

cú pháp: mysql_num_rows(int *result_identifier*) ;

Trong đó : result_identifier là mã số trả về của hàm mysql_query(), mysql_db_query(), mysql_list_tables(), mysql_list_dbs() .

+mysql_affected_rows()

cú pháp : int mysql_affected_rows(int [link_identifier]) ;

Trong đó :int link_identifier là mã số nhận dạng, nó phải được thực hiện trong hàm mysql_select_db() trước đó .

Hàm trả về số dòng đã bị tác động bởi một câu truy vấn SQL :INSERT, UPDATE, DELETE trước đó theo tham số link_identifier. Nếu link_identifier không được chỉ định thì mã kết nối trước đó sẽ được chỉ định.

Chú ý :

- Nếu câu lệnh SQL trước đó là DELETE mà không có mệnh đề WHERE thì toàn bộ các bản ghi trong bảng đã bị xoá nhưng hàm mysql_affected_rows() sẽ trả về gián trị 0.

-Hàm này không có tác dụng đối với câu lệnh truy vấn SELECT. Để lấy được số dòng trả về (số dòng đã bị tác động) bởi câu lệnh SELECT ta dùng

hàm mysql_num_rows().

+mysql_result() : lấy dữ liệu từ result_identifier

cú pháp : mixed mysql_result(int result_identifier, int row, mixed [field]);

Trong đó : result_identifier là mã số trả về của hàm mysql_query(), mysql_db_query(), mysql_list_tables(), mysql_list_dbs() .

row là bản ghi mà ta sẽ lấy dữ liệu

field là trường trong dòng row mà ta sẽ lấy dữ liệu .

Các tham số result_identifier và row phải có, còn tham field là tùy chọn. Hàm sẽ trả lại các nội dung của dòng row và cột field từ tập kết quả được chỉ định bởi biến *result_identifier*. Nếu đối số field không được chỉ định rõ thì trường tiếp theo của bản ghi sẽ được trả về .

Ví dụ:

```
<?php
  $mysql = "select id, name from ds_thanhvien";
  $link = mysql_connect($host, $user, $password);
  $result = mysql_db_query("php", $mysql);
  echo  " mysql_result($result, 0, "id") <BR>\n";
  echo  " mysql_result($result, 0, "name") <BR>\n";
?>
```

+mysql_free_result() : Hàm giải phóng vùng bộ nhớ được liên kết với result_identifier .

cú pháp: mysql_free_result(int result_identifier) ;

Trong đó : result_identifier là mã số trả về của hàm mysql_query(), mysql_db_query(), mysql_list_tables(), mysql_list_dbs() .

Hàm này chỉ được dùng nếu như bạn đánh giá thấy rằng kịch bản của bạn sử dụng quá nhiều bộ nhớ khi đang chạy. Gọi hàm này trên một trình xử lý kết quả sẽ giải phóng toàn bộ dữ liệu liên kết trong bộ nhớ .

Ngoài ra còn các hàm khác:

string **mysql_tablename** (int result_identifier, int i)

Hàm trả lại tên của bảng/csdl tại chỉ số i trong result_identifier.

string **mysql_field_name** (int result_identifier, int field_index)

Hàm trả lại tên của trường tại vị trí field_index trong mã result_identifier

int **mysql_list_dbs** ([int link_identifier])

Hàm trả lại một result_identifier là danh sách biến CSDL trên MySQL Server nếu thành công, lỗi trả về false .

int **mysql_list_tables** (string database [, int link_identifier])

Hàm trả về danh sách tất cả câc bảng trong một CSDL MySQL, thành công trả về một result identifier, giá trị false nếu có lỗi .

int **mysql_list_fields** (string database_name, string table_name [, int link_identifier])

Hàm trả về thông tin liên quan đến một bảng dữ liệu.

int **mysql_num_fields** (int result_identifier)

Trả về số trường trong tập kết quả .

int **mysql_num_rows** (int result_identifier)

Trả về số bản ghi trong tập kết quả, hàm này chỉ có giá đối với các câu lệnh SELECT ,để lấy lại số bản ghi được trả lại từ các lệnh :INSERT, UPDATE hoặc DELETE, dùng mysql_affected_rows().

string **mysql_field_type** (int result_identifier, int field_index)

Hàm trả về kiểu dữ liệu của trường tại vị trí field_index trong mã result_identifier .

int **mysql_field_len** (int result_identifier, int field_offset)

Hàm trả về độ dài của trường được chỉ định thông qua tham số field_offset .

array **mysql_fetch_lengths** (int result_identifier)

trả về một mảng tương ứng với các độ dài của mỗi trường trong bản ghi được lấy về bởi hàm mysql_fetch_row() hoặc false nếu có lỗi.

int **mysql_errno** ([int link_identifier])

Hàm trả về mã lỗi từ hàm thao tác CSDL MySQL trước ,trả về giá trị 0 nếu không có lỗi .

string **mysql_error** ([int link_identifier])

Hàm trả về xâu thông báo lỗi từ hàm thao tác CSDL MySQL trước, trả về xâu rỗng nếu không có lỗi .

object **mysql_fetch_field** (int result_identifier [, int field_offset])

Lấy thông tin về trờng từ tập kết quả rồi trả lại nh một đối tượng.

XI. **Session và Cookie**

1. Session là gì ?

HTTP là giao thức không được xây dựng theo cách để có thể lu giữ được trạng thái giữa hai lần giao dịch. Khi một người dùng yêu cầu truy nhập một trang Web, rồi sau đó người dùng này lại tiếp tục yêu cầu truy nhập đối với trang Web khác thì HTTP không thể biết được rằng đó là hai yêu cầu từ cùng một người dùng.

Ý tưởng của việc điều khiển phiên làm việc là có thể lưu vết của một người dùng trong suốt một phiên làm việc.

Nếu chúng ta làm được điều này thì sẽ dễ dàng cung cấp một truy nhập cho người dùng, từ đó ta có thể lưu vết trạng thái của người dùng và có thể thực hiện việc mua bán trên mạng.

Session trong PHP được điều khiển bởi một giá trị ID duy nhất gọi là "sessionID", giá trị này sẽ được tự động sinh ra và mã hóa. SessionID được sinh ra bởi PHP và được lưu trữ ở phía client trong suốt một phiên giao dịch. Nó có thể được lưu trữ trên các Cookie ở máy người dùng hay truyền lên các URL.

SessionID có tác dụng như một khoá để bạn có thể đăng ký những biến đặc biệt gọi là biến session. Nội dung của những biến này được chứa trên Server. SessionID là những thông tin chỉ thấy được ở phía client. Nếu tại thời điểm nào đó của một kết nối đến trang Web của bạn, sessionID có thể thấy được trên cookie hay URL, bạn có thể truy nhập những biến session chứ trên Server ở phiên làm việc đó.

2) Cookie là gì ?

Cookie là những mẩu tin nhỏ mà trang script có thể chứa trên các máy khách (client). Bạn có thể thiết lập một cookie trên một máy người dùng bằng cách gửi một "HTTP header" có chứa dữ liệu theo dạng sau:

Set-Cookie:Name=VALUE;[expires=DATE;][path=PATH;]

[domain=DOMAIN_NAME;][secure]

câu lệnh này sẽ tạo ra một cookie có tên gọi là NAME với giá trị là VALUE. Trường expires sẽ thiết lập ngày mà cookie sẽ hết hiệu lực, path và domain có thể được sử dụng để chỉ định các URL (nơi mà cookie sẽ được gửi đi). Từ

khoá secure có nghĩa là cookie sẽ không gửi đi trên quá một kết nối HTTP chuẩn.

Khi một browser kết nối tới một URL, trớc tiên nó kiểm tra các cookie đã được lưu trữ trên máy. Nếu có bất kì một cookie nào có liên quan đến địa chỉ URL vừa được kết nối, chúng sẽ được truyền trở lại cho server.

3) Thiết lập các cookie từ PHP

Ta có thể thiết lập các cookie trong PHP bằng cách sử dụng hàm:

int setcookie (string name [,string value [,int expire [,string path [,string domain [, int secure]]]]]);

Những tham đối của hàm tương ứng với những tham đối của Set-Cookie header ở trên.

Nếu ta thiết lập cookie như sau:

setcookie ("TestCookie", "Test Value");

thì khi người dùng đến thăm trang kế tiếp trong site của ta (hoặc reload trang hiện tại) ta sẽ phải truy nhập vào biến với tên là "TestCookie" có chứa giá trị là "Test Value", ta chỉ có thể truy nhập tới nó thông qua biến mảng $HTTP_COOKIE_VARS [] của PHP .

Ta có thể xoá một cookie bằng cách gọi lại hàm setcookie() với tham đối như sau:

tên của cookie là tên của cookie cần xoá và không có trường giá trị.

4) Sử dụng kết hợp cookie với session

Đối với cookie có một số vấn đề sau đây:

Một vài webrowser không thể truy cập được tới các cookie (không hỗ trợ cookie) và một số người dùng không có các cookie trên browser của họ. Đây là lý do để PHP sử dụng cả hai cách thức :cookie và URL method.

Khi sử dụng PHP session, ta sẽ không phải thiết lập các cookie, những hàm session sẽ lưu giữ những thông tin này cho chúng ta.

Để xem nội dung của các cookie đã được thiết lập bởi session ta sử dụng hàm:

session_get_cookie_params(). Hàm này sẽ trả về một mảng liên kết mà các phần tử của mảng chứa các thông tin như: lifetime, path, domain,.. .

Để thiết lập các tham số cho session cookie ta dùng hàm:

void **session_set_cookie_params** (int lifetime [, string path [, string domain]])

5) Lưu giữ sessionID

PHP sẽ sử dụng các cookie mặc định cùng với session. Nếu có thể được, một cookie sẽ được thiết lập chứa SessionID.

Một cách để sử dụng các SessionID trên URL đó là dịch PHP cùng với lựa chọn –enable-tran-sid.

Cách nữa là ta có thể đa sessionID vào trong thẻ link. SessionID được chứa trong hằng SID. Để làm được điều này, ta thêm vào cuối thẻ link hằng SID để dùng nó như là phương thức GET.

Ví dụ:

<A HREF = "link.php?<?=SID?>">

Hằng SID làm việc được như trên chỉ khi ta cấu hình PHP cùng với

--enable-track-vars.

6) Thực thi những phiên làm việc đơn giản.

Những bước cơ bản của việc sử dụng session:

+ Bắt đầu một Session

+ Đăng ký những biến Session

+ Sử dụng biến Session

+ Huỷ bỏ biến Session và kết thúc Session

6.1 Bắt đầu một Session

Cách đơn giản nhất để bắt đầu một Session là dùng hàm:

Bool Session_start();

Hàm này sẽ kiểm tra xem đã có một Session ID nào đã được tạo ra hay chưa. Nếu chưa thì nó sẽ tạo ra một Session ID, còn nếu đã tồn tại một Session ID thì thực chất nó chỉ lấy ra những biến Session để ta có thể dùng nó. Hàm trả về giá trị TRUE nếu thành công ,ngược lại trả về giá trị FALSE.

Chúng ta cũng có thể bắt đầu một Session bằng cách cấu hình PHP để nó tự động bắt đầu khi có ai đó thăm trang Web của ta. Điều này có thể làm được nếu ta chọn session. auto_start trong file c:\Windows\php.ini

Một Session cũng sẽ được bắt đầu khi ta đăng ký một biến Session.

6.2 Đăng ký những biến Session

Để cho một biến có thể lưu dấu thông tin từ một trang script này sang trang script khác, ta cần phải đăng ký nó bằng cách gọi hàm:
bool session_register(mixed VarName [,mixed...]);

Việc đăng ký này sẽ lưu trữ tên biến và ghi giá trị của biến cho đến khi phiên giao dịch kết thúc hoặc khi ta huỷ bỏ (deregister) việc đăng ký biến đó.

Ví dụ:

Để đăng ký biến $Var_name ta viết như sau:

$Var_name= 5;

session_register("Var_name");// không nên sử dụng ký tự $ trong đăng ký một biến

6.3 Sử dụng biến Session

Để đa một biến Session vào trong phạm vi mà nó có thể được sử dụng, ta cần phải khởi tạo một Session bằng một trong những cách đã nêu trên.

Sau đó, ta có thể truy cập được những biến này. Nếu đã đăng ký biến này là toàn cục bằng cách sử dụng hàm register_global(), thì ta có thể truy nhập biến bình thường thông qua tên biến, ví dụ : $Var_name ;
Nếu không khai báo biến là toàn cục thì ta phải truy nhập những biến Session thông qua mảng liên kết $HTTP_SESSION_VARS("Var_name");

Để kiểm tra xem một biến đã được đăng ký là biến Session hay chưa ta dùng hàm: bool session_is_registered (string name);

Hàm này trả về giá trị TRUE nếu biến đã được đăng ký, ngược lại trả về giá trị FALSE.

Ta cũng có thể kiểm tra một biến có là biến Session bằng cách kiểm tra mảng liên kết $HTTP_SESSION_VARS() về sự tồn tại của biến.

6.4 Huỷ bỏ biến Session và kết thúc Session

Khi muốn kết thúc một biến Session, ta có thể huỷ bỏ đăng ký của biến đó bằng hàm : bool session_unregister (string name) ;

Trong đó: name là tên biến ta muốn huỷ đăng ký (tên này không cần có ký tự $)

Hàm trả về giá trị TRUE nếu thành công ,ngược lại trả về giá trị FALSE. Hàm này chỉ có thể huỷ đăng ký của một biến Session tại một thời điểm.

Để huỷ tất cả các biến Session hiện tại, ta dùng hàm: void session _unset ( ) ;

Để kết thúc một Session ta dùng hàm:s

bool session_destroy () ;

Hàm này sẽ xoá đi SessionID và hủy tất cả những dữ liệu liên quan đến Session này. Hàm trả về giá trị TRUE nếu thành công, ngược lại trả về giá trị FALSE.
Ta nên hủy tất cả các biến Session trước khi kết thúc một Session.

CHƯƠNG III : PHƯƠNG PHÁP FAST TEMPLATE TRONG PHP

I. Các kĩ thuật mẫu phổ biến

Các mẫu có thể được dùng nếu như bạn đang thực hiện một site thường xuyên sử dụng lại các thành phần .Trong phạm vi của PHP ,các mẫu ở đây ý nói đến HTML. Không dùng các mẫu ,một web site phức tạp sẽ có cấu trúc rất nặng.

Để giải quyết vấn đề trên ,nhiều kĩ thuật mẫu đã tồn tại, nhúng có thể hữu ích trong một số trường hợp. Chúng ta giới thiệu qua một số kĩ thuật được sử dụng thường xuyên .

Name	Code/ HTML mixed	HTML structure Defined in PHP	Advantages	Disadvantages	Useful for
Embedded PHP	Yes	Yes	Fastand easy	Leadsto unreadable scrips, noreuse of existing code,hardto maintain	Quick and Smallscripts
Separating common parts	Yes	No	Fast and easy, Reuseof certain pasts	Leadsto unreadable scripts, often hard to maintain	Web sites with LOC < 1000
FastTemplate	No	No	Abstracts HTML completely from coding, easy to adapt to new needs	Complex	Web sites with LOC > 1000

Việc sử dụng các mẫu trong các trình ứng dụng

1. Embedded PHP

Chúng ta có thể nhúng các câu lệnh PHP vào trong HTML. Một ví dụ điển hình của cách thức trên :

```
<HTML>
<HEAD>
        <TITLE>powers</TITLE>
</HEAD>
<BODY BGCOLOR=”black” TEXT=”white”>
   <H1>powers</H1>
   <TABLE>
   <TR>
          <TH>i</TH>
          <TH>i^i</TH>
   </TR>
<?php
   for ($i= 0 ; $i< 10 ; $i++) {
   echo “<TR><TD>$i</TD><TD>”.pow($i,$i).”</TD></TR>\n”;
   }
?>
   </TABLE>
   </BODY>
   </ HTML >
```

Phương pháp này được dùng khá phổ biến với những người bắt đầu học lập trình PHP. Tới một lúc nào đó ,những người mới lập trình bằng phương pháp nhúng tiến tới sự thành thạo ở các mức cao hơn, nhưng thật đáng tiếc, cách lập trình này lại ngăn cản điều đó .

2. Separating common parts

Mặc dù kĩ thuật này có sử dụng lại các mã code/HTML và việc thay đổi cũng dễ dàng hơn ,nhưng nó vẫn thừa hưởng hầu hết các nhược điểm của phương pháp PHP nhúng .

Cách làm thông thường là bắt đầu với các HTML header và footer, chúng được viết thành các hàm. Các hàm này có thể được gọi khi cần đến chúng và do đó linh hoạt hơn khi sử dụng. Sự thay đổi các hàm sẽ được phản ánh trong tất cả các trường hợp.

Ta tách ví dụ PHP nhúng ở trên ra hai file .

Tệp prepend.inc chứa hai hàm :hiển thị một HTML header và một HTML footer .

```
<?php        //prepend.inc
function CommonHeader($title) {
?>
<HTML>
<HEAD>
      <TITLE><?php  echo $title  ?></TITLE>
</HEAD>
<BODY BGCOLOR=”black” TEXT=”white”>
<H1><?php  echo  $title  ?></H1>
<?php
}
function CommonFooter() {
?>
</TABLE>
</BODY>
</HTML>
<?php
}
?>
```

\

Tệp main.php sử dụng tệp prepend.inc

```
<?php include "prepend.inc"; ?>
<?php CommonHeader( "power" ); ?>
<TABLE>
<TR>
  <TH>i</TH>
  <TH>i^i</TH>
</TR>
<?php
    for ($i= 0 ; $i< 10 ; $i++) {
       echo "<TR><TD>$i</TD><TD>".pow($i,$i)." </TD></TR>\n";
    }
?>
</TABLE>
<?php CommonFooter(); ?>
```

3. FastTemplate

Trong phần này chúng ta cùng tìm hiểu về phương pháp thiết kế Web trong PHP là FastTemplate. Đây là phương pháp hay, nó được gắn kèm với các gói ứng dụng của PHP, nó được xây dựng trong tệp tin có tên class. FastTemplate.php .

Mục đích của phương pháp này là giúp cho chương trình viết bằng PHP có nhiều giao diện khác nhau trên cùng một cơ sở mã, và làm giảm khối lượng công việc do chương trình được tách ra làm hai phần : phần viết các đoạn mã (code) và phần thiết kế giao diện cho chương trình. Phần viết các đoạn mã sẽ tính toán dữ liệu và thể hiện trên các trang HTML thông qua các biến FastTemplate. Phần giao diện sẽ thiết kế giao diện cho chương trình bằng ngôn ngữ HTML. Trong phần này chúng ta có thể lồng vào trang HTML các ngôn ngữ như Java Applet, JavaScript, ... cho trang web thêm sinh động. Việc thêm vào trang HTML những đoạn mã này không làm ảnh hưởng tới phần đoạn mã PHP. Khi đó, ở vị trí nào trên trang web cần thể hiện các kết quả tính toán từ đoạn mã thì sẽ sử dụng các biến FastTemplate để thể hiện.

II. Phương pháp FastTemplate

1. Biến FastTemplate

Biến FastTemplate là gì ?

Các biến FastTemplate có cấu trúc như sau :

{TEN_BIEN_FASTTEMPLATE}

Biến được đặt trong dấu đóng mở ngoặc nhọn, tên biến ở giữa, tên có thể là chữ hoa hoặc chữ thường. Tuy nhiên trong phần đoạn mã PHP bạn cũng phải viết giống như vậy.

2. Vị trí đặt biến FastTemplate

Các biến FastTemplatse có thể đặt bất kỳ đâu trong trang HTML. Nó có thể là biến thể hiện thông tin trên các trang HTML khi được browser hiển thị hoặc các biến sẽ thêm vào các đoạn mã HTML .

Ví dụ : ta đặt tên tệp là : vidu_template.tpl

```
<HTML>
<HEAD>
<TITLE>{TITLE}</TITLE>
</HEAD>
<BODY>
{CONTENT}
</BODY>
</HTML>
```

Trong ví dụ trên có hai biến FastTemplate là {TITLE} và {CONTENT}. Biến {CONTENT} sẽ thể hiện thông tin trên trang web khi được hiển thị, biến {TITLE} được thêm vào như một thành phần của mã HTML .

3. Một ví dụ minh họa

Theo ví dụ trên chúng ta đã có tệp "vidu_template.tpl", bây giờ chúng ta đi xây dựng đoạn script sau :

```
<?php
include “class.FastTemplate.php”;
# khai báo một đối tượng thuộc lớp FastTemplate
$tpl = new FastTemplate(“.”);
# Định nghĩa một bản đồ các tệp FastTemplate
$tpl->set_filenames(array(“body” => “vidu_template.tpl ”));
# Gán các giá trị cho biến FastTemplate
$tpl->assign_vars(array(
  “TITLE” => “Vi du”,
  “CONTENT” => “Nội dung của trang web” ));
# Cho hiện trang web này
$tpl->pparse(“body”);
?>
```

4. Bốn bước cho một trang web .

Trong một script cần thực hiện bốn bớc theo thứ tự sau :

Bước 1 : Khai báo một đối tượng thuộc lớp FastTemplate

Bước 2 : Gán các tệp chứa mã HTML (tệp này có phần mở rộng không nhất thiết là *.htm) cho các phần tử của mảng được định nghĩa bởi phương thức set_filenames() (bước này được gọi là “định nghĩa một bản đồ các tệp FastTemplate”) .

Bước 3 : Gán giá trị cho các biến FastTemplate trong các tệp .

Bước 4 : Cho hiện trang web bằng phương thức pparse(“tên_trang_web”)

Chú ý :

+ Bước 2: Trong “định nghĩa một bản đồ các tệp FastTemplate” có thể nhiều tệp được gán .

+ Bước 4: Mỗi lần gọi phương thức pparse(); chỉ được một tệp FastTemplate.

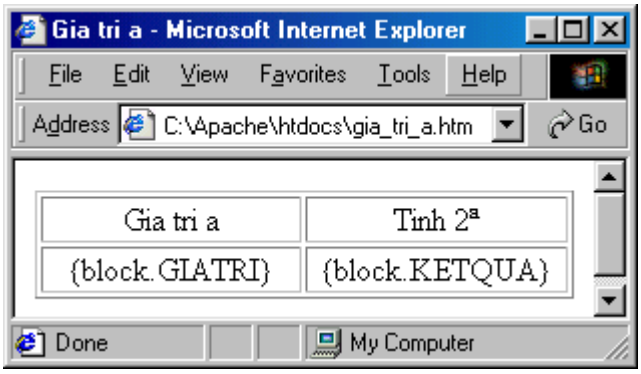
+ Các bước 1,2,3,4 có thể lặp lại nhiều lần .

+ Thứ tự gọi phương thức pparse(); là rất quan trọng vì nó ảnh hưởng đến cách thể hiện của trang web.

5. Khối FastTemplate

Thông tin trên trang web có thể được thể hiện dưới dạng bảng, với số hàng không xác định do đó FastTemplate có hỗ trợ "khối FastTemplate". Khối FastTemplate sẽ lặp đi lặp lại một đoạn HTML nào đó khi được đánh dấu khối. Xét ví dụ sau :

Hãy thực hiện phép tính 2^a với a = 1, ..., n. Kết quả được thể hiện bởi một bảng gồm hai cột, cột giá trị của a và cột kết quả của phép tính 2^a.



Chúng ta tạo tệp "tinh_2_mu.htm"

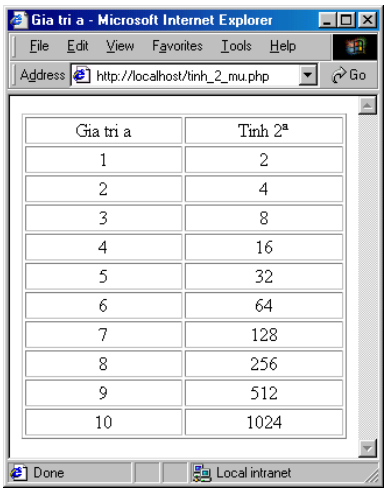
```
<html>
<head>
<title>Tinh 2 mu a</title>
</head>
<body>
<table border="1" width="100%">
  <tr>
    <td width="50%">
    <p align="center">Gia tri a</td>
    <td width="50%">
      <p align="center">Tinh 2ê</td>
  </tr>
  <!-- BEGIN block -->
  <tr>
    <td width="50%">
      <p align="center">{block.GIATRI}</td>
    <td width="50%">
      <p align="center">{block.KETQUA}</td>
```

```
  </tr>
  <!-- END block -->
</table>
</body>
</html>
```

Ta thấy tệp này tạo một bảng hai hàng hai cột, hàng trên là tiêu đề của các cột, hàng dưới cột thứ nhất là giá trị của a và cột thứ hai là kết quả của phép tính 2^a. Hàng dưới được bắt đầu bởi khoá <!-- BEGIN block --> và kết thúc bởi khoá <!-- END block-->. Hai khoá này cho biết phần bên trong sẽ bị lặp (block ở đây là tên khoá), các biến FastTemplate bên trong có dạng {tenkhoa.TEN_BEN}.

Đoạn PHP Script sẽ như sau :

```
<?php
include "class.FastTemplate.php";
$tpl = new FastTemplate(".");
$tpl->set_filenames(array(
   'tinh_2_mu_a' => 'tinh_2_mu.htm'
));
$n = 10;
for ($i = 1 ; $i <= $n ; $i++)
  $tpl->assign_block_vars("block",array(
    'GIATRI' => $i,
    'KETQUA' => pow(2, $i)
  ));
$tpl->pparse("tinh_2_mu_a");
?>
```



Kết quả khi chạy chương trình như hình vẽ.

PHẦN II : HỆ CƠ SỞ DỮ LIỆU MYSQL

CHƯƠNG I. GIỚI THIỆU NGÔN NGỮ MYSQL

I. Giới thiệu chung

MySQL rất phức tạp, nhưng giao diện SQL trực giác và dể học. Trong chương này chúng ta mô tả khái quát về các lệnh, các kiểu dữ liệu và các hàm mà chúng ta cần biến để sử dụng Mysql một cách có hiệu quả và có hiệu suất cao.
Chương này cung cấp tất cả các hàm tham khảo trong Mysql. Để sử dụng chương này có hiệu quả bạn có thể tìm đến hàng loạt các bảng đã được liệt kê.
Các ưu điểm của MySQL:

- MySQL là một hệ quản trị nhỏ, bảo mật, và rất dễ sử dụng, thường được sử dụng cho các ứng dụng nhỏ và trung bình. Nó được sử dụng cho các ứng dụng client / server với máy chủ mạnh như UNIX, Windows NT và Windows 95/98, và đặc biệt trên máy chủ UNIX .
- MySQL hỗ trợ các điểm vào là ANSI SQL92 và ODBC mức 0-2 SQL chuẩn.
- MySQL hỗ trợ nhiều ngôn ngữ cho việc thông báo lỗi như : Czec, Dutc, English, Estonian, French, German, Hungarian, Italian, Norwegian Nynorsk, Polish, Portuguese, Spanish and Swedish. Ngôn ngữ được hỗ trợ mặc định cho dữ liệu là ISO-8859-1 (Latin1), muốn thay đổi phải sửa trong mã nguồn .
- Ngôn ngữ lập trình sử dụng viết các hàm API để thâm nhập cơ sở dữ liệu MySQL có thể là C, Perl, PHP.. .
- Các bảng (table) trong cơ sở dữ liệu MySQL có kích thước rất lớn và được lưu ở thư mục Datas. Kích thước lớn nhất của một bảng tối thiểu là 4GB và nó còn phụ thuộc và kích thước lớn nhất của một file do hệ điều hành quy định .

- Cơ sở dữ liệu MySQL rất dễ quản lý và có tốc độ xử lý cao hơn tới ba bốn lần so với các hệ quản trị cơ sở dữ liệu khác.
- MySQL là một hệ quản trị cơ sở dữ liệu mô hình quan hệ, nó có mã nguồn mở. Nó được cung cấp miễn phí trên các máy chủ UNIX, OS/2 và cả trên Windows.

Bên cạnh các ưu điểm trên MySQL cũng có một vài nhược điểm :

- MySQl không cho phép thực hiện các câu lệnh SQL select truy vấn con. Ví dụ : SELECT deptno, ename, sal

  FROM emp x

  WHERE sal > (SELECT AVG(sal)

  FROM emp

  WHERE x.deptno = deptno)

  ORDER BY deptno
- MySQL không hỗ trợ Stored Procedures, Triggers, Transactions, Foreign Keys, và Views như các hệ quản trị cơ sở dữ liệu khác .

II .Cấu trúc ngôn ngữ

1. Đưa ra một xâu và một số như thế nào

Mục này mô tả những cách khác nhau để đưa ra xâu và số trong Mysql. mục này cũng bao gồm nhiều sắc thái và ``gotchas'' điều mà bạn có thể gặp khi đi sâu vào các kiểu cơ bản trong Mysql.

2. Xâu

Xâu là một dãy kí tự, được đặt trong dấu nháy đơn ‘ ’ hoặc dấu nháy kép “ ” (nếu chạy trong phương thức ANSI thì xâu chỉ được đặt trong dấu nháy đơn)

Ví dụ: ‘trời nắng’.

“chào buổi sáng”.

Trong một xâu, có sự kết hợp của các kí tự tạo nên các kí tự đặc biệt. Mỗi kí tự đặc biệt này bắt đầu bằng dấu chéo ngược (\ ) được xem như kí tự escape .

Mysql nhận biết các kí tự đặc biệt sau:

\0 : kí tự 0 trong mã ASCII.

\’ : Kí tự nháy đơn ‘.

\" : Kí tự nháy kép “.
\ b : Kí tự Backspace.
\n : Kí tự xuống dòng mới.
\r : Kí tự về đầu dòng .
\t : Nhẩy ô (tab).
\z : Kí tự 26 trong mã ASCII. Kí tự này có thể được mã hoá để đứng ở cuối file trong Windows.
\\ : Kí tự chéo ngược \ .
\% : Kí tự % .
\ _ : Kí tự gạch dưới _ .
chú ý : nếu như sử dụng kí tự ‘\%’ hoặc kí tự gạch dưới ‘\ _’ trong một số trường hợp có thể trả về \% và \ _ chứ không phải là % và _ .
* có một số cách để xâu gồm cả dấu nháy :

- để một dấu nháy đơn (‘) nằm trong một xâu được đánh dấu bởi dấu nháy đơn thì phải viết là: ‘ ‘ ’ ’ .
- để một dấu nháy kép (“) nằm trong một xâu được đánh dấu bởi dấu nháy đơn thì phải viết là: ‘ “ ” ’ .
- có thể đặt kí tự (‘\’) trước kí tự đánh dấu .
- một dấu nháy đơn trong cặp dấu nháy kép đưa ra xâu chứa dấu nháy đơn, một dấu nháy kép trong cặp dấu nháy đơn đưa ra xâu chứa dấu nháy kép.
- Ví dụ :

```
mysql> SELECT 'hello', '"hello"', '""hello""', 'hel"lo','\'hello';
-----------------------------------------
| hello | "hello" | ""hello"" | hel'lo | 'hello |
-----------------------------------------

mysql> SELECT "hello", '"hello"', ""hello"", "hel""lo", "\"hello";
-----------------------------------------
| hello | 'hello' | "hello" | hel"lo | "hello |
-----------------------------------------

mysql> SELECT "This\nIs\nFour\nlines";
--------------------
| This
Is
Four
lines |
--------------------
```

nếu bạn muốn chèn dữ liệu nhị phân vào trong cột BLOB. Những kí tự sau phải được thể hiện dưới dạng những kí tự đặc biệt :

- NUL : ASCII 0. phải được thể hiện ‘ \0 ’ .
- \ : ASCII 92. dấu gạch ngược phải được thể hiện ‘ \\ ’ .
- ’ :ASCII 39. dấu nháy đơn phải được thể hiện ‘ \’ ’ .
- ” : ASCII 34. dấu nháy kép phải được thể hiện ‘ \” ’.
- nếu biết mã C, có thể sử dụng hàm CAPI là Mysql-escape-string() để ESC kí tự bằng câu lệnh INSERT.
- Nên sử dụng hàm ESC đối với những xâu chứa kí tự đặc biệt được liệt kê ở trên.

3. Số (Numbers)

Sô bao gồm số nguyên và số thực .

Số nguyên được biểu diễn bằng một dãy số. Còn số thực sử dụng dấu chấm để phân cách phần thập phân .

Dấu gạch dưới được đặt trước số để chỉ số âm.

Ví dụ:

```
1221
0
32
```

Ví dụ cho số thực:

```
94.42
32032.6809e10
48.00
```

chú ý : số nguyên được sử dụng trong ngữ cảnh số thực được chuyển thành số thực có giá trị ngang bằng

ví dụ : float 0 → 0.0

float 56 -> 56.0

Những giá trị hệ Hexa (hệ 16)

Mysql hỗ trợ các giá trị hệ Hexa. Trong ngữ cảnh số thì chúng giống như một số nguyên. Trong ngữ cảnh xâu thì chúng giống như một xâu nhị phân mà mỗi cặp số nguyên Hexa được chuyển thành một kí tự .

Ví dụ:

```
mysql> SELECT x'FF'
-> 255
mysql> SELECT 0xa0;
-> 10
mysql> select 0x5061756c;
-> Paul
```

ở ví dụ trên khi viết một số dưới dạng Hexa bạn phải điền vào vào trước số đó chuỗi 'x' hoặc '0x'. Cú pháp x 'xâu Hexa' dựa trên ANSISQL, và cú pháp 0x dựa trên ODBC. Xâu hệ Hexa thường sử dụng ODBC để đưa giá trị cho cột BLOB Có thể chuyển một xâu hoặc số thành hệ Hexa bằng hàm HEX().

4. Giá trị NULL

Giá trị NULL có nghĩa là "không có dữ liệu"("no data") khác với giá trị 0 đối với kiểu số và xâu rỗng đối với kiểu xâu .

Trong khuôn dạng xuất hoặc nhập File dữ liệu NULL có thể được thể hiện bằng '\N' (như LOAD DATA INFILE, SELECT...INTO OUTFILE).

5. Tên của Cơ sở dữ liệu, bảng, chỉ số cột và bí danh

Tên của cơ sở dữ liệu, bảng, chỉ số cột và bí danh tất cả đều tuân theo một qui luật chung trong Mysql. Nhưng chú ý rằng qui luật này đã được thay đổi bắt đầu với Mysql Version 3.23.6 khi chúng ta dưa vào lời chú thích cho việc nhận biết cơ sở dữ liệu, bảng và tên cột với dấu ' ' ' ' .' " 'sẽ làm việc nhận ra lời chú thích nếu bạn chạy trong chế độ ANSI .

Định danh (Identifier)	Độ dài	Các kí tự cho phép
Database (cơ sở dữliệu)	64	Tất cả các kí tự cho phép trong thư mục ngoại trừ '/' và '.'
Table (Bảng)	64	Tất cả các kí tự cho phép với tên File ngoại trừ '/' và '.'
Column (Cột)	64	Cho phép tất cả các kí tự
Alias (Bídanh)	256	Cho phép tất cả các kí tự

Chú ý: ở bảng trên định danh không thể có kí tự đánh dấu, ASCII(0) hoặc ASCII(255) .

Nếu định danh là một từ bị hạn chế hoặc trong đó có chứa các kí tự đặc biệt thì bạn phải sử dụng dấu nháy đơn để chú thích khi sử dụng chúng .

Ví dụ: SELECT * from `select` where `select`.id > 100;

Trong phiên bản trước của Mysql, luật đặt tên như sau :

- Một tên gồm các kí tự trong bảng mã, cũng cho phép kí tự gạch dưới '_' và '$'.
- Tên có thể bắt đầu bởi tất cả các kí tự hợp pháp, trong thực tế tên có thể bắt đầu bằng một số (điều này khác nhiều với hệ cơ sở dữ liệu)tuy nhiên tên không chỉ gồm một số .
- Không được sử dụng kí tự '.' Trong tên bởi vì kí tự đó được sử dụng để mở rộng khuôn dạng để truy nhập đến cột (xem ở bảng dưới đây).

Chúng ta không nên dùng những tên như 1e bởi vì một biểu thức như 1e1thì rất mơ hồ, nó có thể được biên dịch là một biểu thức 1e1 hoặc là số 1e1.

Trong Mysql ta có thể liên hệ tới một cột bằng việc sử dụng những cách sau:

Cột tham chiếu	ý nghĩa
Col_name	Cột col_name từ bảng sử dụng trong câu truy vấn chứa cột với tên này.
Tbl_name.col_name	Cột col_name từ bảng tbl_name của cơ sở dữ liệu hiện thời .
Db_name.tbl_name. col_name	Cột col_name từ bảng tbl_name của cơ sở dữ liệu Db_name.
'column_name'	Một cột có chứa kí tự đặc biệt hoặc từ khoá

Bạn cần phải chỉ rõ tbl_name hoặc db_name.tbl_name đối với những cột có cùng tên trong nhiều bảng của cơ sở dữ liệu, hoặc bảng có cùng tên trong các cơ sở dữ liệu khác nhau để tránh sự nhập nhằng .

Ví dụ : giả sử mỗi bảng T1 ,T2 chứ một cột C bạn có thể lấy C trong một lệnh SELECT sử dụng cả T1 và T2. Trong trường hợp này C nhập nhằng bởi vì nó không duy nhất giữa các bảng được sử dụng trong câu lệnh. Bạn phải chỉ ra tên bảng bằng cách viết T1.C hoặc T2.C. tương tự nếu bạn lấy từ một bảng T trong cơ sở dữ liệu DB1 và từ bảng T trong cơ sở dữ liệu DB2, bạn phải tham chiếu đến cột trong những bảng đó như sau : DB1.T.col_name và DB2.T.col_name.

Cú pháp .Tbl_name có nghĩa là bảng Tbl_name nằm trong cơ sở dữ liệu hiện thời .cú pháp này được chấp nhận cho tương thích ODBC bởi vì một số chương trình ODBC thêm vào đầu tên bảng một dấu chấm '.'

6. Phân biệt chữ hoa và chữ thường đối với tên trong Mysql :

Trong Mysql tên cơ sở dữ liệu, bảng trong Windows thì không phân biệt chữ hoa và chữ thường, còn trong Unix thì có phân biệt chữ hoa và chữ thường .
Chú ý : mặc dù trong windows không phân biệt chữ hoa và chữ thường, bạn cũng không nên dùng tên cơ sở dữ liệu, bảng với các trường hợp khác nhau trong cùng một câu truy vấn .
Ví dụ :

```
mysql> SELECT * FROM my_table WHERE MY_TABLE.col=1;
```

câu lệnh trên sẽ không thực hiện được .
Tên cột không phân biệt chữ hoa và chữ thường trong tất cả các trường hợp .
Bí danh trên bảng phân biệt chữ hoa và chữ thường
Ví dụ:

```
mysql> SELECT col_name FROM tbl_name AS a
    WHERE a.col_name = 1 OR A.col_name = 2;
```

Câu lệnh trên sẽ không thực hiện được vì bí danh phân biệt 'a' và 'A'
Bí danh trên cột sẽ không phân biệt chữ hoa và chữ thường.
Nếu bạn muốn truy vấn thực hiện được trên bảng thì chấp nhận một qui ước luôn luôn tạo ra tên cơ sở dữ liệu và tên bảng sử dụng chữ thường.
Một cách tránh vấn đề này là sẽ bắt đầu mysql với
-0 lower_case_table_names = 1, Theo mặc định tùy chọn này là 1 trên Windows và 0 trên Unix. Nếu lower_case_table_names là 1 MySQL sẽ chuyển đổi tất cả các tên bảng về chữ thường ,Chú ý rằng nếu bạn thay đổi tùy chọn này, bạn cần trước hết chuyển đổi những tên bảng cũ của bạn về chữ thường trước khi bắt đầu mysql.

7. Biến người sử dụng

Mysql hổ trợ các biến riêng cùng với cú pháp @variable name. Tên biến bao gồm kí tự anpha tư tập các kí tự và thêm vao kí tự '_', '$', và '.'
Biến không phải khởi tạo, NULL không xuất hiện trong biến, biến có thể chứa số nguyên thực hoặc giá trị xâu. Tất cả các biến sẽ tự giải phóng vùng nhớ khi chúng thoát ra.
Bạn có thể gán biểu thức vào biến theo cú pháp : @variable:=expr
Ví dụ:

```
select @t1:=(@t2:=1)@t3:=4,@t1,@t2,@t3;
----------------------------------------------
| @t1:=(@t2:=1)@t3:=4 | @t1  | @t2  | @t3  |
----------------------------------------------
|                   5 |    5 |    1 |    4 |
----------------------------------------------
```

chúng ta phải sử dụng cú pháp:= để gán biến, bởi vì dấu bằng (=) được sử dụng để so sánh .

biến có thể được sử dụng trong một biểu thức lưu ý rằng điều này không được sử dụng trong ngữ cảnh một số được yêu cầu rõ ràng như trong mệnh đề LIMIT của lệnh select hoặc mệnh đề IGNORE number LINES của lệnh LOAD DATA .
chú ý : trong lệnh SELECT mỗi biểu thức chỉ nhận một giá trị khi nó được gửi cho người dùng. Điều này cũng đúng trong mệnh đề HAVING, GROUP BY, ORDER BY. Bạn không thể tham chiếu tới một biểu thức gồm tập hợp các biến trong SELECT.
Ví dụ:
SELECT (@aa:=id) AS a, (@aa3) AS b FROM table_name HAVING b=5;
Câu lệnh trên không thực hiện đượcbởi vì @aa sẽ không chứa giá trị ở hàng hiện tại trừ phi gí trị của id thay cho cột đã được chấp nhận trước .

8. Các dòng chú thích

Mysql cho phép chèn các câu chú thích vào trong kịch bản của Mysql, khi thực hiện trình thông dịch sẽ bỏ qua tất cả các đoạn văn bản là câu chú thích
Trong Mysql chú thích được đặt sau dấu : # dòng chú thích
-- dòng chú thích
cách viết này chỉ có tác dụng trên một dòng .
dùng cặp kí hiệu sau : /* ....*/ để chèn các câu chú thích, loại chú thích này được gọi là chú thích khối .
ví dụ: mysql> select 11; # chú thích trên một dòng
mysql> select 11; -- chú thích trên một dòng
mysql> select 1 /* chú thích khối*/ 1;
mysql> select 1
/*
chú thích trên nhiều dòng
chú thích khối
*/
1;
lưu ý rằng chú thích – yêu cầu bạn phải có ít nhất một dấu cách sau --!
Mặc dù vậy có vài hạn chế trong cách ghi chú *..*\ đó là:

- kí tự nháy đơn và nháy đôi dùng để chỉ sự bắt đầu của một xâu thậm chí sử dụng bên trong một chú thích nếu trong chú thích dấu nháy thứ nhất không phù hợp với dấu nháy thứ hai thì hệ phân tích sẽ không nhận ra điểm kết thúc của ghi chú.
- dấu chấm phẩy (;) được dùng để báo kết thúc của một câu lệnh Mysql và sau dấu (;) là sự bắt đầu của câu lệnh tiếp theo.

CHƯƠNG II. CÁC THÀNH PHẦN CỦA MY SQL

Trong phần này chúng ta giới thiệu khái quát về các kiểu dữ liệu của các trường va các thuộc tính về kiểu trong mỗi loại dữ liệu.Sau đó là một số kiểu bảng trong MySQL và cuối cùng là một số hàm cơ bản được cung cấp bởi MySQL.

I. Các kiểu dữ liệu

Trong MySQL có các kiểu dữ liệu sau:

Kiểu số
Kiểu kí tự (char,varchar,Blob,Text,Enum,Set)
Kiểu xâu.
Kiểu ngày giờ (Datetime,Date,TimeStamp,Time,year)

1. Kiểu số

Kiểu tinyint:
Khai báo TINYINT[Cm] [UNSGNED] [ZEROFIU].Đây là kiểu số nguyên với giá trị rất nhỏ:

-Nếu khai báo với từ khoá nusigned thì giá trị mà cột có kiểu này có thể nằm trong khoảng 0 đến 255.

-Nếu không có từ khóa nusiged thì giá trị trong khoảng –128 đến 127

Kiểu Smallint
Khai báo:Smallint [Cm] [nusiged] [zerofill].Đây là kiểu số nguyên có giá trị nhỏ:

-Nếu là số có dấu thì giá trị nằm trong khoảng-32768 đến 32767

-Nếu là số không dấu thì giá trị trong khoảng 0 đến 65536.

Kiểu Mediumint
Khai báo :Mediumint [cm] [nusiged] [zerofill] [(m)]

-Nếucó dấu thì giá trị nằm trong khoảng –8388608 đến 8388607.

-Nếu là số có dấu thì giá trị nằm trong khoảng 0 đến 16777215.

Kiểu Int:
Khai báo:Int:[Cm] [nusiged] [zerofill]

-Với số có dấu giá trị trong khoảng –2147483648 đến 2147483647

-Với số không dấu giá trị trong khoảng 0 đến 429467295

Kiểu Interger
Khai báo:interger [Cm] [nusiged] [zerofill].Tương tự kiểu INT.

Kiểu Bigint
Khai báo :Bigint [Cm] [nusiged] [zerofill].Đây là kiểu nguyên mà giá trị là rất lớn.

-Với số có dấu giá trị trong khoảng –9223372036854775808 đến 9223372036854775807.

-Với số không dấu giá trị trong khoảng 0 đến 18446744073709551615

Note:tất cả các phép toán số học sử dụng số có dấu bigint hoặc double, vì vậy không sử dụng số không dấu ,mà giá trị vượt quá số nguyên lớn tức là nếu nhân hai số nguyên lớn thì giá trị trả về sẽ vượt quá số nguyên lớn có dấu.
Kiểu float:
Khai báo:float (precision)[zerofill].Đối với số thực độ chính xác đơn thì độ chính xác <24.Đối với số thực độ chính xác kép thì độ chính xác <255.Float(x) giống kiểu double,float nhưng cỡ và số chữ số phần thập phân chưa được định nghĩa

Precision:số chữ số phần thập phân mà float có thể nhận.Float [Cm,d] [zerofill]:đây là số thực với độ chính xác kép với m hiển thị độ rộng,d là số các chữ số phần thập phân.

Nếu dùng float không có đối hoặc đối <=24,thì tương ứng dùng số thực có độ chính xác đơn.
Double [(Cm,d)] [zerofill]:số thực với độ chính xác kép với m là độ rộng, d là số chữ số phần thập phân.Nừu dùng double không đối hoặcdùng float(x) với 25<=x<=53 thay cho một số thực độ chính xác kép.
Double precision [(Cm,d)][zerofill]
Real [(Cm,d)][zerofill] .tương tự như số double.
Decimal [(Cm,d)][zerofill] được coi như một kiểu char,được lưu trữ như một xâu mỗi kí tự là một kí số của giá trị cần lưu trữ.
Nuneric [(Cm,d)][zerofill] tương tự decimal.

2.Kiểu ngày giờ

2.1 Datime, Date, TimeStamp.
Các kiểu Datime, Data, TimeStamp có những đặc trưng riêng nhưng cũng có mối liên quan.
Có một số Y2K chúng ta đã từng biến đổi với các kiểu thời gian không gian rõ ràng trên

MySQL đã giải thích các ngày cùng với giá trị năm mập mờ theo quy luật sau:

-Giá trị năm trong khoảng 70-69 được chuyển sang năm 2000-

2069

-Giá trị năm trong khoảng 70-99 được chuyển sang năm 1970-1999.

Đó là vì giá trị năm ta chỉ để hai chữ số, còn hai số trước mặc định là năm 1900.Để làm cho không nhầm lẫn năm thì phảI cung cấp giá trị năm là bốn số.MySQL đã làm điều này.

Kiểu datetime: Đây là kiểu ngày giờ được dùng khi bạn cần lưu trữ thông tin cả về ngày tháng và thời gian.Giá trị của nó nằm trong khoảng 1000-01-10:00:00 đến '9999-12-3123:59:59'

MySQL hiển thị datatime theo định dạng:

"yyyy-mm-dd hh:mm:ss" nhưng nó cũng cho phép ấn định giá trị cho kiểu này bằng dạng số hoặc xâu.

Kiểu Date được dùng khi bạn chỉ cần lưu trữ giá trị ngày tháng.

MySQL truy xuất và hiển thị giá trị DATE với định dạng "yyyy-mm-dd"

Giá trị trong khoảng '1000-01-01' đến '9999-12-31'có thể ấn đinh bằng số hoặc xâu.

Kiểu TIMESTAMP [Cm]

Cung cấp 1 kiểu mà bạn có thể dùng để đánh dấu một cách tự động đối với phép INSERT hoặc UPDATE cùng với ngày giờ hiện tại.Nếu có nhiều cột TIMESTAMP thì chỉ cột đầu tiên được cập nhật một cách tự động.

Để cập nhật vào cột đầu như vậy cũng phải thoả mãn các điều kiện sau:

-Cột mà chứa được xác định chi tiết trong mệnh đề INSERT hoặcLOADDATAINFILE

-Cột mà không xác định chi tiết trong lệnh UPDATE và một số cột khác đã thay đổi giá trị

-Bạn đặt chi tiết cột TIMESTAMP là ngày giờ hiện tại bằng cách ấn định giá trị NULL

Các cột khác có thể được đặt thời gian và ngày tháng hiện tại.Chỉ cà với cột giá trị NULL hoặc NOW.Giá trị của TIMESTAMP là từ 1970-01-01:00:00:00.Đến năm 2037 thì MySQL hiển thị TIMESTAMP theo định dạng 'yyyy mm dd hh mm ss'

'yyyy mm dd hh mm ss'.MySQL cũng cho phép sử dụng bằng xâu hoặc số.

Kiểu TIME

MySQL định dạng giá trị thời gian là 'hh:mm:ss' hoặc HHH:MM:SS Nguyên nhân mà phần giờ có thể lớn hơn tại vì kiểu TIME không chỉ biểu diễn thời gian trong 1ngày (giờ phải <24) mà nó còn là khoảng thời gian giữa hai sự kiện (có thể >24 hoặc thậm chí là âm).Giá trị của

nó trong khoảng '-838:59:59' tới '838:59:59'

Kiểu YEAR

Có thể định dạng bởi hai hoặc bốn chữ số, giá trị nằm trong khoảng 1901 đến 2155 với bốn chữ số.

Với hai số thì mặc định từ 00-69 là 2000-2069 ,70 đến 99 là 1970-1999

3.**Kiểu kí tự (String)**

Kiểu char: Độ dài cột char là cố định khi bạn tạo bảng. Giá trị độ dài trong khoảng 0-255 nếu số kí tự điền vào nhỏ hơn độ dài quy định thì nó sẽ tự động điền vào kí tự trống.

Trong MySQL kiểu char cho phép bạn tạo một cột có dạng char(0),điều này cũng thuận tiện trong trường hợp bạn phải tuân theo với số phần mềm cũ nó phụ thuộc và sự tồn tại của một cột mà không thực sự cần sử dụng giá trị. Cũng tốt trong trường hợp bạn cần định nghĩa một cột sao cho nó có thể nhận hai giá trị một cột khai báo char(0) nó có thể chiếm một bit và có thể nhận hai giá trị NULL hoặc " "

Kiểu Varchar [national] varchar(m)[binary]. Đây là cột được khai báo với một xâu có độ dài không cố định những kí tự trống sẽ được huỷ khi lưu trữ giá trị, m nằm trong khoảng 1-255.

value	Char(4)	Storage required	Varchar	Storage required
' '	' '	4bytes	' '	1bytes
'ab'	'ab'	4bytes	'ab'	3bytes
'abcd'	'abcd'	4bytes	'abcd'	5bytes
'abcdefgh'	'abcd'	4bytes	'abcd'	5bytes

Kiểu Blob và TEXT

-BLOB là đối tượng nhị phân lớn có thể chứa lượng rất lớn các dữ liệu với 4 kiểu BLOB:TINYBLOB, BLOB, MEDIUMBLOB và LONGBLOB.Có 4 kiểu TEXT là TINYTEXT,TEXT, MEDIUMTEXT, và LONGTEXT

TINYBLOB,TINYTEXT:có độ rộng tối đa là255

Blob và Text độ dài tối đa là 65535 kí tự.

Mediumblob,mediumtext:độ dài tối đa 16777125 kí tự.

Longblob,longtext: độ dài tối đa 424967295 kí tự.

Chú ý: Mô hình client/server giới hạn bởi 16Mb. Một gói giao tiếp trên một hàng nên không thể sử dụng toàn bộ phạm vi của kiểu này.
Kiểu ENUM(value1,value2,..)(kiểu liệt kê). Một đối tượng xâu chỉ có thể có 1 giá trị chọn từ một danh sách các giá trị
‘’value1”,”value2”,..null hoặc kí tự đặc biệt.

Một kiểu liệt kê có giá trị lớn nhất là 65535 giá trị.
Kiểu tập hợp set(value1,value2,..) một đối tượng xâu có thể nhận 0 hoặc nhiều hơn mỗi loại có thể chọn từ danh sách các giá trị :”value1”,”value2”, .. Một tập hợp có tối đa 64 thành phần.
Giá trị null nghĩa là không có dữ liệu,nó khác giá trị 0 và xâu rỗng.

Giá trị null có thể được kí hiệu bởi \n cho việc truy xuất file dạng text.

4 .Cột chỉ số.

Tất cả các kiểu cột của MySQL có thể được chỉ số hoá .Việc dùng chỉ số hoá trên các cột có liên quan với nhau là cách tốt nhất để cải tiến việc thực hiện của câu lệnh Select.

Một bảng có thể có tới 16 cột chỉ số, độ dài chỉ số tối đa là 256 bytes ,mặc dù có thể thay đổi khi biên dịch MySQL.

Đối với cột kiểu char và varchar bạn có thể chỉ số hoá phần đầu của cột.Như vậy sẽ nhanh hơn và cần ít không gian đĩa hơn là chỉ số hoá trong cả cột.Cú pháp dùng trong mệnh đề CREATE TABLE như sau:

KEY index_name(colname(length)).

Ví dụ sau sẽ tạo một chỉ số hoá cho 10 kí tự đầu của cột tên:

```
MySQL>CREATE  TABLE test(
Name char(200)not NULL
KEY index_name(name(10)))
```

Đối với cột kiểu BLOB và TEXT, bạn phải chỉ số hoá một phần đầu của cột, không thể chỉ số hoá cả cột được.

Tạo chỉ số trên nhiều cột.

MySQL có thể tạo chỉ số trên nhiều cột. Một việc chỉ số hoá có thể lên tới 15 cột(trên cột char và varchar bạn có thể chỉ số hoá phần đầu của cột như một phần của việc chỉ số hoá).

Việc chỉ số hoá nhiều cột được xem như một mảng được sắp xếp mà chứa các giá trị được tạo bởi việc kết nối các giá trị giữa các cột được chỉ số hoá .

Mysql sử dụng chỉ số hoá nhiều cột như là cách để truy vấn nhanh hơn khi bạn đã xác định được số lượng lớn thông tin của cột đầu tiên của sắp xếp trong một câu lệnh WHERE, thậm chí nếu bạn

không xác định giá trị đối với các cột khác thì thông tin lấy được vẫn chính xác .

Ví dụ :

```
Mysql> CREATE TABLE test(
            Id int NOT NULL,
            Last_name char(30)NOT NULL,
            First_name char(30)NOT NULL,
            PRIMARY KEY(id),
            INDEX name(Last_name, First_name);
```

Khi đó tên sẽ được chỉ số thông qua 2 cột Last_name, First_name. Việc chỉ số sẽ được áp dụng cho việc truy vấn đến các giá trị cụ thể đó trong phạm vi của cột last_name hoặc cả Last_name và First_name. Do đó, cột tên sẽ được dùng trong truy vấn dưới đây:

```
Mysql>SELECT* FROM test WHERE
last_name=”Lan”;

Mysql>SELECT* FROM test WHERE last_name=”Lan”
                AND first_name=”Nguyễn”;

Mysql>SELECT* FROM test WHERE last_name=”Lan”

AND(first_name=”Nguyễn”)OR(first_name=”Lê”)

Mysql>SELECT* FROM test WHERE last_name=”Lan”;

AND(first_name>=”L”)AND(first_name<=”M”)
```

Tuy nhiên cột tên sẽ không được dùng trong truy vấn dưới đây :

```
Mysql>SELECT* from test WHERE last_name=”Lan”
Mysql>SELECT* from test WHERE last_name=”Lan”
                OR(first_name=”Nguyễn”)
```

Sử dụng các kiểu cột từ những bộ máy CSDL khác

Đây là cách tạo ra một cách dễ dàng có sử dụng mã SQL cho các hệ quản trị CSDL khác có cơ sở là SQL.Đây chính là ánh xạ kiểu từ CSDL khác sang kiểu tương ứng bên MySQL,do đó dễ dàng chuyển bảng từ CSDL khác sang bảng có kiểu trong MySQL .

Kiểu ở các thành phần khác	Kiểu trong mySQL
Binary(num)	Char(num)binary
Char varyfing(num)	Varchar(num)
Float4	Float
Float8	double
Int1	Tinyint
Int2	Smallint
Int3	Mediumint
Int4	Int
Int8	Bigint
Long	Varbinaryint mediumBlob
Long	Varchar mediumText
Midleint	Mediumint
Varbinary	Varchar(num)Bynary

Nếu bạn tạo bảng với kiểu sử dụng là của bộ máy CSDL khác thì khi đưa ra lệnh :DISCRIBE tbl_name,khi đó Mysql sẽ ghi lại cấu trúc bảng này với kiểu tương ứng trong Mysql.

5.Các kiểu bảng trong MySQL

Trong Mysql có 4 kiểu bảng khác nhau ,có thể tạo nhiều kiểu bảng với câu lệnh

CREATE TABLE:

Kiểu bảng MyISAM(indexsequential acces method):Đây là kiểu bảng mặc định được tạo ra khi bạn dùng lệnh Create,nó là cơ sở trong mã lệnh ISAM và có một sô sự mở rộng hữu dụng tạo ra file với phần mở rộng .MYI

Kiểu bảng MyHeap:Các bảng kiểu này được lưu trữ trong bộ nhớ ,việc chỉ số hoá của chúng được thực hiện bởi hàm băm,vì vậy mà tốc độ thực hiện của bảng kiểu này cực kì nhanh,nhưng nó phảI trả giá rất đắt ,vì dữ liệu của ta bị mất .Vì vậy bảng này chỉ dùng tạm thời ,bạn có thể chỉ ra sô hàng tối đa trong câu lệnh Create table .Kiểu bảng này có thể chiếm gần hết bộ nhớ của bạn ,vì vậy không thể sử dụng các cột kiểu blob,text hoặc auto_increment.

Bảng kiểu BDB:Các bảng kiêu này là những giao dịch an toàn ,chúng được cung cấp các lệnh COMMIT,ROOLBACK,các bảng kiểu này hoạt động chậm hơn các bảng có kiểu MyISAM

Bảng kiểu Merge:Đây là kiểu bảng mới ,mã lệnh cho kiểu bảng này mới đang ở phiên bản bêta,tuy nhiên nó sẽ nhanh chóng ổn định ,đây chính là bảng ISAM nhưng chúng chỉ được chọn lựa một bảng ,chỉ được insert,update,delete một bảng mà bạn chọn ,nếu bạn muốn xoá bảng chỉ cần chỉ định xoá merge.Bảng kiểu này giúp xử lý các vấn đề sau ,dễ quản lý và tách ra thành nhiều bảng khác nhau ,nâng cao tốc độ xử lý ,việc tìm kiếm sẽ hiệu quả hơn.

II. Các hàm trong MySQL

Hàm dùng trong câu lệnh SELECT và WHERE

1. Hàm nhóm.

```
Mysql>select 12*3
    ->7
Mysql>select (12)*3
    ->9
```

2. Các phép toán xử lý thông thường

```
Cộng : Mysql>select 35
    ->8
Trừ :  Mysql>select 3-5
    ->-2
Nhân: Mysql>select 3*5
    ->15
Chia:Mysql>select 3/5
    ->0.6
  Mysql>select 3/0
    ->NULL
```

3. Phép toán thao tác với bit

Số bit tối đa là 64 bits

Phép AND

```
        Mysql>select 29&15;
    ->13
```

Phép OR:

```
    Mysql>select 29\15;
    ->31
```

Dịch tráI:

```
    Mysql>select 1<<2;
    ->4
```

Dịch phải:

Mysql>select 4>>2;
->1

4. Phép toán lôgic

Các phép and(&),or(::),not(!).Các giá trị trả về là 0 hoặc 1 tương ứng đúng hoặc sai

Ví dụ: Mysql>select not 1;
->0

5. Các phép so sánh(>=,>,<=,<)

MySQL thực hiện phép so sánh theo quy tắc sau:

-Nếu một trong hai đối số là NULL thì kết quả so sánh là NULL,trừ phép tương đương .

-Nếu cả hai đối số là kiểu xâu thì được so sánh như giữa các xâu với nhau .

-Nếu cả hai đều là số nguyên thì được so sánh như các số nguyên.

-Giá trị hexa được xem như là xâu nhị phân nếu không so sánh với số khác .

-Nếu một trong những đối số là kiểu TIMESTAMP hoặc DATETIME còn đối số kia là một hằng số thì hằng số đó sẽ phảI được chuyển sang kiểu TIMESTAMP trước khi so sánh.Đó chính là làm cho việc sử dụng ODBC thân thiện hơn.

-Trong tất cả các trường hợp khác thì đối số được xem như là số thực.

Ví dụ:

Mysql>SELECT 1>'6x';
->0
Mysql>SELECT 7>'6x';
->1
Mysql>SELECT 1='x6';
->1
Mysql>SELECT 1 is NULL,0 IS NULL,NULL IS NULL
->0 0 1

phép toán BETWEEN:

expr BETWEEN min AND max
Mysql>select 1BETWEEN 2 AND 3;
->0
Mysql>select 'b'BETWEEN 'a' AND 'c';
->1
Mysql>select 2BETWEEN 2 AND 'x-3';
->0

Phép toán in,not in(value1,…)

```
mysql>select 2 IN (0,4,'dfdg');
    ->0
mysql>select 'dfdg' IN (0,4,'dfdg');
    ->1
```

Phép toán ISNULL

```
mysql>select ISNULL(11);
    ->0
mysql>select ISNULL(1/0);
    ->1
```

6. Các hàm so sánh xâu

Đối với các hàm so sánh xâu thì có phân biệt chữ hoa ,chữ thường

expr LIKE pat[ESCAPE 'escape-char']

giá trị trả về có thể là số hoặc xâu,với hàm này có thể dùng hai kí tự đại diện là %,-

%: giá trị trả về có thể là số các kí tự

-: giá trị phù hợp chỉ là một kí tự

Tương tự ta sẽ có hàm NOT LIKE: NOT (**expr LIKE pat[ESCAPE 'escape-char']**)

Ví dụ :

```
mysql>select 'hà' LIKE 'hà\-';
->1
mysql>select 10 LIKE '1%';
->1
```

Hàm STRCMP(expr1,expr2):trả về 0 nếu 2 xâu như nhau ,-1 nếu đối số đầu nhỏ hơn đối số hai theo thứ tự sắp xếp ,ngoàI ra thì bằng 1

```
mysql>select STRCMP ('text1','text2');
->-1
mysql>select STRCMP ('text1','text');
->1
mysql>select STRCMP ('text1','text1');
->0
```

Phép toán cast

```
mysql>select "a"="A";
    ->1
mysql>select BINARY "a"="A";
    ->0
```

Hàm IF(expr1,expr2,expr3)

Tương đương với hàm if kép trong ngôn ngữ C,nếu expr1 đúng thì hàm nhận giá trị expr2,ngược lại thì nhận giá trị expr3

Ví dụ:

```
Mysql>select IF(1<2,4,5);
->4
Mysql>select IF(1>2,yes,no);
    ->no
```

giá trị trả về có thể là kiểu sô hoặc xâu tuỳ theo ngữ cảnh phù hợp .

7. Các hàm toán học.

-ABS(x): giá trị tuyệt đối của x

```
mysql>select ABS(-3);
    ->3
mysql>select ABS(3);
    ->3
```

-SIGN(X): Dấu của biểu thức trong ngoặc ,trả lại 0 nếu X=0,-1 nếu X<0;1 nếu X>0

```
mysql>select (3-5);
    ->-1
mysql>select (0);
    ->0

mysql>select (3);
    ->1
```

-MOD(N,M):lấy phân dư của phép chia n cho m

```
mysql>select MOD(30,6);
    ->0
mysql>select 13%6;
    ->1
```

-ROUND(X):Lấy phần nguyên gần X nhất

-exp(x),log(x),log10(x)

-POW(x,y):x mũ y

Tương tự ta có các hàm toán học khác như:

sin(x),cos(x),tan(x),asin(x),acos(x),atan(x),rand();

8. Các hàm xử lý xâu:

-Char(N,…) hàm trả về dãy các kí tự có mã ACCSI được liệt kê trong ngoặc

VD: mysql>select CHAR(77,121,83,81,'76');

->'MySQL'
mysql> select CHAR(77,77.3,'77.3');
->'MMM'

-Hàm ghép 2 xâu :
CONCAT(str1,str2,…)
vd : mysql> select CONCAT('MY','S','QL');
->'MYSQL'
mysql> select CONCAT('MY',NULL,'QL');
->NULL
mysql> select CONCAT(12.5);
->12.5
-CONCAT_WS(separator,str1,str2..)
mysql> select CONCAT_WS(",","first name","last name");
->'first name,last name'

mysql> select CONCAT_WS(",","first name",NULL,"last name");
->'first name,last name'
-Các hàm về độ dàI của xâu :
LENGTH(str)
0CTET_LENGTH(str)
CHAR_LENGTH(str)
CHARACTER_LENGTH(str)
Mysql>select LENGTH('text');
->4
mysql>select 0CTET_LENGTH('text')
-Các hàm định vị xâu con
LOCATE(substr,str)
POSITION(substr IN str)
Mysql>select LOCATE('nam','phươngnam');
->7
Mysql>select LOCATE('nama','phươngnam');
->0

NgoàI ra trong Mysql còn rất nhiều các hàm liên quan đến việc xử lý xâu khác nhưng không liệt kê ở đây .Ta sẽ sang chương mới

CHƯƠNG III. CÁC LỆNH THAO TÁC VỚI CSDL

I. Các lệnh thao tác với CSDL

1. Lệnh tạo CSDL

Cú pháp :CREATE DATABASE[if not exists] db_name;

Tạo ra CSDL như là tạo một thư mục mà trong đó có các file,đây chỉ là tạo thư mục chứ chưa có file,mà các file ở đây là các bảng.Vậy ta sẽ phải có câu lệnh tạo bảng

CREATE DATABASE IF NOT EXISTS NHANSU;

Câu lệnh trên sẽ tạo ra một CSDL tên NHANSU,nhưng chưa có bản nào được tạo ra.

Từ khoá if not exists nếu được chỉ ra sẽ tránh được một lỗi phát sinh nếu như trên máy đã có cơ sở dữ liệu này rồi.

2. Lệnh xoá CSDL.

Cú pháp:Drop Database [if esists]db_name.

Câu lệnh này sẽ xoá tất cả các bảng trong cơ sở dữ liệu và xoá luôn cả cơ sở dữ liệu có tên là DB_NAME.

Ví dụ:Drop Database [if eists] NHANSU

Nếu bạn thực hiện xoá một cơ sở dữ liệu với kết nối cơ sở dữ liệu đó thì cả CSDL nguồn và kết nối sẽ bị xoá.Bạn phảI rất cẩn thận đối với lệnh này,nó sẽ xoá thư mục với tên của CSDL và tất cả các file với các phần mở rộng như sau:

.BAK	.DAT	.HSH	.ISD
.ISM	.ISM	.MRG	.MYD
.MYI	.DR	.FRM	

Trong các phiên bản 3.22 hoặc cũ hơn ta có thể chỉ định từ khoá if esists để ngăn một lỗi xuất hiện nếu như CSDL chưa có trên đĩa.

II. Các lệnh thao tác trên bảng và xử lý đối với bảng.

1. Lệnh tạo bảng.

Cú pháp:

```
CREAT{TEMPORARY}TABLE{IF NOT ESISTS}
        Tbl_name [(creat_definition,…)]
        [table_options] [select_statement]
```

Lệnh sẽ tạo ra một bảng trong cơ sở dữ liệu hiện tại nếu cơ sở dữ liệu chưa tồn tại thì sẽ có lỗi phát sinh.Trong phiên bản Mysql3.2.2 hoặc muộn hơn tên bảng có thể được chỉ định rõ là db_name.table_name trong trường hợp này máy sẽ thực hiện mà không quan tâm tới việc cơ sở dữ liệu có tồn tại hay không.
Trong đó:
Tbl_name là tên bảng cần tạo
table_options :chỉ ra kiểu của bảng cần tạo và những đặc tính của bảng cần tạo
table_options ={ISAM | MYISAM | HEAP | MERGE}
or AUTO_INCREMENT=#:Giá trị tiếp theo mà bạn muốn đặt cho bảng
or AUTO_ROW_LENGTH=#:Giá trị trung bình độ dàI của hàng trong bảng
or COMMENT="string":lời bình luận tối đa 60 kí tự
or MAX_ROWS=# số hàng tối đa mà bạn định lưu trữ
creat_definition:được định nghĩa như sau:
creat_definition:
col_name type [NOT NULL | NULL] [DEFAULT default_value]
{ AUTO_INCREMENT}
[PRIMARY KEY}{reference_definition]
or PRIMARY KEY (index_col_name,…)
or KEY [index_name] (index_col_name,…)
or INDEX [index_name] (index_col_name,…)
or UNIQUE {INDEX} [index_name] (index_col_name,…)
or FULLTEXT {INDEX} [index_name] (index_col_name,…)
or {CONSTRANINT symbol] FOREIGN KEY index_name ((index_col_name,…)
[reference_defintion]
or CHECK (expr)
type:Kiểu của cột được tạo trong bảng
REFERENCES tab_name [(index_col_name,…)]
[MATCHFULL | MATCHPARTIAL]
[ON DELETE reference_option]
[ON UPDATE reference_option]
reference_option:
RESTRICT | CASCADE | SET NULL | NO ACTION | SET DEFAULT

Trong phiên bản 3.23 bạn có thể sử dụng từ khoá temporary khi tạo một bảng,bảng tạm này sẽ tự động được xoá khi kết thúc một phiên làm việc .
NOT NULL: Nếu cột được chỉ định là NOT NULL thì khi nhập liệu ta bắt buộc phảI nhập dữ liệu cho cột này .
NULL :Đối với cột kiểu timestamp sẽ có sự khác biệt về giá trị NULL so với những cột có những kiểu khác,bạn không thể đưa ra giá trị NULL với cột có kiểu timestamp nếu đặt là NULL thì nó sẽ tự động đặt ngày giờ hiện tại,bởi vì cột có kiểu stamp sử dụng cách thức này nên thuộc tính NULL hoặc NOT NULL sẽ không được đặt như những cột thông thường và nó sẽ được lờ đI nếu bạn chỉ định đIều này .
DEFAULT<giá trị ngầm định >:cột này sẽ tự động được đặt là giá trị ngầm định nếu như ta bỏ qua không nhập giáv trị cho cột này .Giá trị ngầm định phảI là hằng số theo nghĩa là ta không thể đặt giá trị mặc định cho cột là giá trị trả về của một hàm ,nếu cột được khai báo là NOT NULL thì giá trị ngầm định sẽ phụ thuộc kiểu của cột
Nếu không đặt DEFAULT
-AUTO_INCREMENT:chỉ có đối với cột số nguyên cột này sẽ tự động được tăng khi nhập dữ liệu .Nếu bạn xoá một hàng chứa giá trị lớn nhất của cột auto_increment thì giá trị này sẽ dùng lại cho lần nhập liệu sau (đIều này chỉ xảy ra đối với bảng loại ISAM mà không xảy đối với bảng MYISAM)
chú ý:Mỗi hàng chỉ có một cột đặt là auto_increment và phảI được đặt là chỉ số
-UNIQUE KEY:đây là khoá của bảng nó chỉ có thể nhận một giá trị nhất định,sẽ có lỗi nếu như ta chèn vào một hàng của bảng giá trị khoá trùng với khoá của hàng đã tồn tại .
-PRIMARY KEY:Định nghĩa khoá chính của bảng,là khoá duy nhất với ràng buộc rằng tất cả các giá trị khoá đều phảI đặt NOT NULL mỗi bảng chỉ có thể có một khoá chính.
-FOREIGN KEY :khoá ngoại dùng làm khoá chính của bảng khác
-Mệnh đề SELECT :Nếu có thêm mệnh đề SELECT sau câu lệnh CREATE TABLE thì Mysql sẽ tạo các trường mới cho tất cả các thành phần nằm trong câu lệnh SELECT

Ví dụ ta tạo bảng hosinh

```
CREATE TABLE HOCSINH(,
  Mahs char(5) NOT NULL,PRIMARY KEY,
  Hoten varchar(35),
  Ngaysinh date,
  Quequan varchar(40));
```

Bây giờ ta tạo bảng SINHVIEN gồm có một trường masv và các trường hoten,quequan,ngaysinh lấy từ bảng HOCSINH:

```
CREATE TABLE SINHVIEN(,
        Masv char(5) NOT NULL PRIMARY KEY,
        SELECT hoten,ngaysinh,quequan from HOCSINH);
```

Mỗi bảng tbl_name sẽ được thay bởi một số file trong thư mục cơ sở dữ liệu ,nếu kiểu bảng là MYISAM bạn sẽ thu được các file sau khi thực hiện lệnh tạo bảng

Tên file	Chức năng
Tbl_name.frm	file định dạng bảng
Tbl_name.Myd	file chứa dữ liệu
Tbl_name.MyI	file chỉ số

2. Lệnh sửa cấu trúc bảng

Cú pháp :ALTER[IGNORE] TABLE tbl_name
alter_spec[,alter_spec...] alter_specification :
ADD[COLUMN] create_definition[FIRST |AFTER colum_name]
Or ADD[COLUMN] (create_definition, create_definition,..)
Or ADD INDEX[index_name] (index_col_name,..)
Or ADD PRIMARY KEY(index_col_name,..)
Or ADD UNIQUE[index_name](index_col_name,..)
Or ADD FULLTEXT[index_name](index_col_name,..)
OR ADD[CONSTRAIN symbol] FOREIGN KEY index_name
(index_col_name)
[referent_definition]
or ALTER[COLUM] col_name{SET DEFAULT literal |DROP DEFAULT}
OR CHANGE[COLUM] old_col_name create_definition
Or MODIFY[COLUMN] create_definition
OR DROP[COLUMN] column_name
OR DROP PRIMARY KEY
OR DROP INDEX index_name
Or RENAME[TO] new_tbl_name
Or table_options

Lệnh ALTER TABLE cho phép bạn sửa cấu trúc của một bảng đã có,ví dụ bạn có thể thêm hoặc xoá cột,tạo hoặc huỷ chỉ số,thay đổi kiểu của cột đã có ,hoặc đổi tên cột hoặc đổi tên bảng ,bạn có thể thay đổi lời chú thích cho bảng hoặc kiểu của bảng .

Nếu sử dụng lệnh ALTER TABLE để thay đổi một cột đã được chỉ định rõ nhưng DESCRIPE tbl_name chỉ ra rằng cột không thể thay đổi ,có nghĩa là Mysql bỏ qua sự thay đổi này do một lý do nào đó .Ví

dụ nếu bạn cố thay đổi một cột có kiểu varchar sang cột kiểu char thì Mysql sẽ không thực hiện việc thay đổi nếu trong bảng có các cột có độ dài thay đổi .
Lệnh ALTER TABLE thực hiện việc tạo một bản sao tạm thời của bảng nguồn việc sửa đổi thực hiện trên bảng sao này ,bảng nguồn sẽ được xoá khi bảng mới được đổi tên vì hoạt động theo cách này nên việc cập nhật sẽ được thực hiện một cách tự động một lần nữa tới bảng mới

Để sử dụng lệnh ALTER TABLE bạn cần phải có quyền :select,insert,update,create,drop đối với bảng.
-Một số ví dụ dùng lệnh ALTER TABLE :

Để sửa đổi tên một cột :

ALTER TABLE hocsinh CHANGE hokhautt,hokhautt varchar (40);
Nếu bạn muốn thay đổi kiểu của cột mà không muốn đổi tên thì bạn vẫn phải viết tên của cột 2 lần .

ALTER TABLE hocsinh CHANGE hokhautt,hokhautt CHAR (40);
Ta cũng có thể dùng MODIFY để thay đổi kiểu của cột

ALTER TABLE hocsinh modify ngaysinh char(10);
Nếu sử dụng lệnh CHANGE hoặc modify với một cột đã sắp xếp mà file chỉ số đã tồn tại thì bạn không thể sắp xếp nhiều hơn số kí tự đã được chỉ số hoá .
Khi dùng change hoặc modify Mysql sẽ cố gắng chuyển dữ liệu sang kiểu mới một cách tốt nhất .
-drop index :huỷ file chỉ số nếu cột được sắp xếp lôgic (chỉ số) mà ta xoá nó thì nó cũng được xoá ở thành phần chỉ số này cũng sẽ được xoá
Drop primary key xoá khoá chính
-ORDER BY:cho phép tạo một bảng mới với hàng được chỉ ra ở mệnh đề order by ,chú ý rằng bảng sẽ không còn như cũ trong mệnh đề order sau khi chèn và xoá
Đổi tên bảng : ALTER TABLE old_name rename new_name;

để chuyển một cột từ kiểu integer sang tinyint not null(với cùng tên) và đổi cột có tên là hoten char(10)->char(20) và đổi từ hoten->hovaten
ALTER TABLE sinhvien modify masv tinyint not null ,change hoten hovaten char(20);
Bổ sung thêm trường ngaysinh
ALTER TABLE sinhvien add ngaysinh timestamp ;
để chuyển cột thành khoá chính

ALTER TABLE sinhvien add primary key(masv);

3. Lệnh đổi tên bảng

Cú pháp :Rename table tbl_name |to newtbl_name |,tbl_name2 to newtablename,…]

Việc đổi tên thực hiện một cách tự động có nghĩa là trong khi thực hiện đổi tên sẽ không thể truy nhập vào bảng này .

Việc đổi tên thực hiện từ trái qua phải ,vì vậy nếu bạn muốn chuyển đổi giữa các bảng bạn sẽ thực hiện như sau :

```
RENAME TABLE old_table to backup_table,
                    New_table to old_table,
                    Backup_table to new_table;
```

Bạn có thể đổi tên bảng giữa 2 CSDL khác nhau :

RENAME TABLE curent_database.table_name to otherdatabasse.table_name

Khi bạn thực hiện câu lệnh rename bạn không thể thực hiện việc khoá bảng hoặc thực hiện một giao dịch.

4. Lệnh xóa bảng:

Cú pháp :Drop table [if esists] tbl_name1 [,tbl_nme2,...] [restrict/cascade]

Lệnh này có thể xoá một bảng hoặc nhiều bảng,tất cả dữ liệu trong bảng sẽ bị xoá.Lệnh Drop table không phảilà một giao diện an toàn.

5. Lệnh tối ưu bảng.

Cú pháp:

Optimize table tbl_name [,tbl_name,…]

Lệnh này nên dùngkhi bạn xoá phần lớn dữ liệu của bảng hoặc bạn thực hiện nhiều sự thay đổi đến bảng như thay đổi độ dàI của các hàng(Đối với bảng có kiểu varchar,bolb),hoặc xoá bản ghi đã được duy trì trong một danh sách liên kết và câu lệnh insert sau đó dùng lại vị trí của bản ghi vừa xoá.

Lệnh này chỉ thực hiện với bảng kiểu MYISAM và BOB.

6. Lệnh Check table (Kiểu tra bảng)

Cú pháp:

Check table tbl_name [tbl_name,…][option…];

Option=Quick/fast/medium/extend/changed

Câu lệnh trên chỉ thực hiện đối với bảng MYISAM nếu không chỉ định option nó sẽ mặc định là medium.Câu lệnh thực hiện việc kiểm tra lỗi của bảng.

7. Lệnh tạo bảng sao lưu(dự trữ)

Cú pháp:Backup table tbl_name [,tbl_name…] to
'path/to/backup/directory';
Nó sẽ thực hiện việc sao chép toàn bộ các file của bảng tới thư mục lưu trữ,kích thước lưu trữ là nhỏ nhất.

8. Lệnh phục hồi bảng đã Backup.

Cú pháp:Restore table tbl_name,…]
From'/path/to/backup/directory'
Phục hồi một bảng đã được sao lưu chỉ thực hiện đối với bảng đã được backup bảng đã tồn tại sẽ không bị ghi đè, bạn sẽ gặp khó lỗi nếu có phục hồi một bảng đã tồn tại.

9. Lệnh phân tích bảng.

Cú pháp:analiyze table tbl_name [,tbl_name …]
Phân tích và lưu thuộc tính khoá cho bảng trong khi phân tích bảng sẽ bị khóa với khoá read lệnh trên chỉ thực hiện đối với bảng MYISAM,BOB.MYSQL sử dụng việc sao lưu các thuộc tính khoá để quyết định trình tự việc kết nối giữa các bảng khi thực hiện việc kết nối với một ràng buộc nào đó.

10. Lệnh repair table.

Cú pháp:REPAIR TABLE tbl_name1 [,tbl_name2,…]
[Quick] [extended]
Lệnh trên chỉ thực hiện đối với bảng MYISAM,câu lệnh này tương tự đối với việc chạy myisamchk_ r table_name;
Lệnh thực hiện sửa chữa bảng bị hỏng.

11. Lệnh Delete.

Delete [low_priority] from tbl_name
[where <ĐIều kiện>]
[limit rows] //giới hạn số hàng cần xoá.
Thực hiện việc xoá các bản ghi từ tbl_name mà nó thoả mãn đIều kiện ràng buộc của mệnh đề where và trả lại số hàng bị xoá.
Nếu không có mệnh đề where thì tất cả các hàng sẽ bị xoá nếu muốn biết số bản ghi bị xoá bạn sẽ thực hiện theo lệnh sau:DELETE FROM tbl_name WHERE1>0;

12. Lệnh Truncate.

Cú pháp: Truncate table tbl_name ;

Câu lệnh trên tương tự như lệnh delete from tbl_name;
Nhưng nó cũng có một số khác biệt:

-Nó sẽ xoá bảng và tạo lại bảng mới với cấu trúc của bảng bị xoá,vì vậy nó sẽ thực hiện nhanh hơn là xoá nhiều hàng.

-Nó không phảI là một giao dịch an toàn:nó sẽ tự động kết thúc giao dịch hiện thời nếu lệnh commit được gọi.

-Không trả lại số dòng đã bị xoá.

13. Lệnh Select.

Cú pháp:

```
SELECT{STRAIGHT_JOIN}{SQL_SMALL_RESULT}
{SQL_BIG_RESULT}
{SQL_BUFFER_RESULT}
{HIGH_PRIORITY}
{DISTINCT | DISTINCTROW |ALL}
Select_expression,...
[INTO {OUTFILE | DUMPFILE}'file_name'export_options]
[FROM table_references
 [WHERE where_definition]
 [GROUP BY {unsigned_integer |col_name | formula}]
[HAVING  where_definition ]
[ORDER BY {unsigned_integer |col_name | formula}[ASC
|DESC],..]
[LIMIT{ofset,}rows]
[PROCEDURE procedure_name]]
```

Đây là lệnh dùng để lấy dữ liệu từ các hàng của một hoặc nhiều bảng dữ liệu.Đây là câu lệnh thao tác chính của ngôn ngữ SQL
Một số ví dụ về câu lệnh MySql

```
CREATE TABLE persons(
Id SMALLINT UNSIGNED NOT NULL
AUTO_INCREMENT,
Name char(60) NOT NULL,
PRIMARY KEY (id));
CREATE TABLE shirts(
Id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT,
Style ENUM('t_shirt','polo','dress') NOT NULL,
Color ENUM ('red','blue','orange','white','black') NOT NULL,
Owner SMALLINT UNSIGNED NOT NULL REFERENCES persons,
PRIMARY KEY (id));

INSERT INTO persons VALUES  (NULL,'antonio Paz');
```

```
 INSERT INTO shirts VALUES
(NULL,'polo','blue',LAST_INSERT_ID()),
(NULL,'dress','white'LAST_INSERT_ID()),
(NULL,'t_shirt,'blue'LAST_INSERT_ID());
INSERT INTO persons VALUES(NULL,'Lilliana Angelovska');

INSERT INTO shirts VALUES
(NULL,'dress','orange',LAST_INSERT_ID()),
(NULL,'polo','red'LAST_INSERT_ID()),
(NULL,'dress,'blue'LAST_INSERT_ID());
(NULL,'t_shirt,'white',LAST_INSERT_ID());
```

SELECT * FROM persons;

Id	Name
1	Antonio Paz
2	Lilliana Angelovska

SELECT * FROM shirt;

id	style	color	Owner
1	Polo	Blue	1
2	Dress	White	1
3	T_shirt	Blue	1
4	Dress	Orange	2
5	Polo	Red	2
6	Dress	Blue	2
7	T_shirt	white	2

```
SELECT s.* from persons p,shirts s
WHERE p.name LIKE 'LILIANA%'
AND s.owner=p.id
AND s.color<>'white';
```

Id	style	color	Owner
4	Dress	Orange	2
5	Polo	Red	2
6	dress	blue	2

14. Các câu lệnh cập nhật dữ liệu

a. Chèn dòng mới vào bảng

Câu lệnh **Insert** :

INSERT [LOW_PRIORITY / DELAYED / IGNORE]
[INTO] tbl_name [(col_name,.. .)]
VALUES (giá trị cột,.. .), (...),.. .
hoạc :
INSERT [LOW_PRIORITY / DELAYED / IGNORE]
[INTO] tbl_name [(col_name,.. .)]
SELECT.. .
hoạc :
INSERT [LOW_PRIORITY / DELAYED / IGNORE]
[INTO] tbl_name
SET col_name_1=giá trị cột 1, col_name_2=giá trị cột 2,.. .
Trong đó :

+Nếu sử dụng thành phần LOW_PRIORITY câu lệnh chỉ có thể insert dữ liệu khi không có ai đọc dữ liệu từ bảng này, và người sử dụng phải đợi khi câu lệnh kết thúc .

+Nếu sử dụng thành phần DELAYED ngừi sử dụng sẽ không phải đợi cho tới khi câu lệnh insert kết thúc mà việc thêm dòng mới vẫn đợi khi không có ai đọc dữ liệu từ bảng này. Thành phần này còn cho phép thực hiện câu lệnh theo từng khối (thực hiện nhiều câu lệnh insert cùng lúc).

+Thành phần IGNORE sẽ bỏ qua các lỗi khi thêm dòng mới vào bảng. Trường hợp lỗi thường xẩy ra khi trùng dữ liệu là khoá, dòng mới sẽ không được thêm vào bảng .

Ví dụ :

```
INSERT INTO author (id, fullname, email)
        VALUES ('j001','John Writer','jw@somewhere.nice.com');
```

hoạc :

```
INSERT INTO author SET
id='j001',fullname='JohnWriter',email='jw@somewhere.nice.com';
```

hoạc :

```
INSERT INTO tempauthor (id, fullname, email)
SELECT id, fullname, email FROM author
        WHERE id >= '100' ;
```

b. Câu lệnh Replace

Cú pháp :

REPLACE [LOW_PRIORITY / DELAYED]
[INTO] tbl_name [(col_name,.. .)]
VALUES (giá_trị_cột,.. .)
hoạc :
REPLACE [LOW_PRIORITY / DELAYED]
[INTO] tbl_name [(col_name,.. .)]
SELECT.. .
hoạc :
REPLACE [LOW_PRIORITY / DELAYED]
[INTO] tbl_name
SET col_name1=giá_trị_cột1, col_name2=giá_trị_cột2,.. .

Câu lệnh này không giống với câu lệnh INSERT , nó thực hiện như sau:

+Một dòng (bảng ghi) mới được tạo ra nếu cột khoá không bị trùng hoạc bảng không có cột khóa.

+Nếu trùng cột khoá toàn bộ dữ liệu không phải là khoá có trong danh sách cột sẽ được thay bằng giá trị tương ứng.

+Nếu toàn bộ dữ liệu cần thay đổi mà bị trùng một dòng (bảng ghi) thì dữ liệu sẽ không được thêm vào.

c. Câu lệnh Update

Câu lệnh thay thế một hoặc nhiều cột của một dòng (bảng ghi) thoả mãn điều kiện nào đó .

Cú pháp :

UPDATE [LOW_PRIORITY] tbl_name SET col_name1=gía_trị1, col_name2,.. .
[WHERE biểu_thức_điều_kiện] [LIMIT #]

+Câu lệnh này làm thay đổi giá trị cột của một dòng (bảng ghi) nếu thoả mãn điều kiện chỉ ra trong mệnh đề WHERE.

+Nếu không có mệnh đề WHERE toàn bộ các dòng sẽ bị thay đổi .

+Chúng ta có thể giới hạn dòng (bảng ghi) bị thay đổi bằng tuỳ chọn LIMIT.

Ví dụ :

UPDATE document SET title='Table of contents',

comment='Fixed typo in the title' WHERE id=321;

Câu lệnh trên cập nhật lại trường title và comment trong bảng document với mã id = 321.

III. Các Thao tác tên bảng

1. Lựa chọn các hàng

- Bạn có thể chọn các hàng riêng biệt từ bảng của mình.Nếu bạn muốn kiểm tra lại sự thay đổi mà bạn đã thực hiện với ngày sinh của browser ,chọn bản ghi browser như sau :
  Mysql>SELECT*FROM pet WHERE name="browser"
- Kết quả xác nhận rằng năm được ghi đúng bây giờ là năm 1989 mà không phải là năm 1998
- Các xâu sử dụng rất phong phú .Do đó bạn có thể sử dụng tên là "browser" hay "BROWSER" thi kêt quả không thay đổi.
- Bạn có thể định rõ giới hạn trên bất kỳ cột nào (khác cột tên) .Ví dụ nếu bạn muốn biết có những loài động vật nào sinh sau năm 1998 ta sẽ kiểm tra cột ngày sinh:
  Mysql>SELECT * FROM pet WHERE both>="1998-1-1";
- Bạn có thể kết hợp các đk để xác định những con chó cái
  Mysql> SELECT * FROM pet WHERE species="dog" AND sex="f";

  Ta co thể dùng các toán tử "AND" hoặc "OR" để kết hợp các đk
  Mysql> SELECT * FROM pet WHERE species="snake" OR sex="bird";
- "AND" và "OR" có thể được trộn lẫn nhau ,nếu thực hiện điều đó tốt hơn hết hãy sử dụng dấu ngoặc đơn.
  Msql> SELECT * FROM pet WHERE (species = "cat" AND sex = "m"> OR (species = "dog" AND sex = "f");

2. Lựa chọn các cột:

- Nếu bạn không muốn xem toàn bộ bảng mà chỉ xem những cột mà bạn đang quan tâm
  .Ví dụ nếu bạn muốn biết các con vật của mình được sinh khi nào ,hãy chọn các côt tên và ngày sinh:
  Msql> SELECT name, birth FROM pet;
- Để tìm ra ai sở hữu những con vẹt ,sử dụng câu lệnh sau:
  Mysql> SELECT owner FROM pet;

- Tuy nhiên ,chú ý rằng câu lệnh chẳng qua là lấy lại trường người sở hữu từ mỗi bản ghi,và một vài trong số chúng xuất hiện nhiều hơn một lần.Để các giá trị không bị lăp lại ta thêm từ khoá DISTINCT:

  Mysql> SELECT DISTINCT owner FROM pet;
- Bạn có thể sử dụng diều kiện WHERE để lựa cjọn các dòng cùng với các cột.Ví dụ để xem ngày sinh của chó và mèo thôi thì ta làm như sau:

  Msql> SELECT name, species, birth FROM pet
  ->WHERE species = "dog" OR species = "cat";

3. Phân loại các hàng

- Có thể bạn đã chú ý tới các ví dụ trước đây và thấy các hàng kết quả được hiển thị không theo một thứ tự đặc biệt nào .tuy nhiên thường dễ dàng hơn khi kiểm tra kết quả khi hàng đã được phân loại theo một cách có ý nghĩa .Để phân loại một kết quả ta sử dụng mệnh đề " ORDER BY " sau đây là ví dụ về ngày sinh của các con vật được phân loại theo ngày:

  Mysql> SELECT name, birth FROM pet ORDER BY birth;
- Để phân loại theo thứ tự ngược lại ,ta thêm từ khoá "DESC" Vào tên cột mà bạn đang phân loại :

  Mysql> SELECT name, birth FROM pet ORDER BY birth DESC;
- Bạn có thể sắp xếp trên nhiều cột khác nhau .Ví dụ để sắp xếp theo kiểu vật sau đó là ngày sinh mà không quan tâm đến kiểu con vật đồng thời các con vật trẻ nhất xếp đầu tiên ,ta sử dụng câu lệnh sau;

  mysql> SELECT name, species, birth FROM pet ORDER BY species, birth DESC;

Chú ý rằng từ khoá "DESC" chỉ áp dụng với tên cột ở ngay trước nó

4. Xác định ngày tháng ,năm cho các kết quả tính toán:

- MySQL cung cấp một rất nhiều chức năng mà bạn có thể sử dụng để thực hiện các kết quả tính toán theo ngày.Vi dụ để tính toán tuổi hoặc trích dẫn một phần của ngày. Để xác định mỗi con vẹt của bạn là bao nhiêu tuổi ,tính toán tuổi như là khoảng cách giữa ngày sinh và ngày hiện tại. Điều đó được thực hiện như sau:

  mysql> SELECT name, (TO_DAYS(NOW())-TO_DAYS(birth))/365 FROM pet;

- Mặc dù các câu lệnh đang làm việc nhưng có một vài điều về nó vẫn có thể cải tiến .
- Đầu tiên kết quả có thể được xem dễ dàng hơn nếu các hàng được trình bày theo một thứ tự nào đó
- Thứ hai, tiêu đề cho cột tuổi thì không có ý nghĩa lắm. Vấn đề thứ nhất có thể được giải quyết bằng cách thêm vào từ khoá "ODER BY" để sắp xếp kết quả theo tên .Để liên hệ với tiêu đề của cột, cung cấp một tên cho cột với một nhãn xuất hiện trong kết quả(được gọi là bí danh)

```
mysql> SELECT name, (TO_DAYS(NOW())-
TO_DAYS(birth))/365 AS age
    -> FROM pet ORDER BY name;
```

- Nếu muốn sắp xếp kết quả theo tuổi hơn là theo tên, ta chi việc sử dụng một câu lệnh "ODER BY" khác

```
mysql>  SELECT name, (TO_DAYS(NOW())-
TO_DAYS(birth))/365 AS age
   ->FROM pet ORDER BY age;
```

- Một câu lệnh có thể được sử dụng để xác định tuổi thọ cho những con vật đã chết. Bạn xác định các con vật nào được kiểm tra có hay không giá trị chết là NULL. Sau đó đối với các con có giá trị non_NULL, tính toán sự khác nhau giữa giá trị chết và sinh.

```
mysql>  SELECT name, birth, death, (TO_DAYS(death)-
  TO_DAYS(birth))/365 AS age
        ➔ FROM pet WHERE death IS NOT NULL ORDER BY age;
```

name	birth	Death	age
Bowse	1989-08-31	1995-07-29	| 5.91

- Câu lệnh sử dụng "death is NOT NULL" vẫn tốt hơn là "death=!=NULL" bởi vì NULL là giá trị đặc biệt. Điều này được giải thích ở phần sau ,
- Nếu bạn muốn biết con vật nào có ngày sinh vào tháng sau ?với kiểu tính toán này ,năm và ngày đều không có liên quan ,bạn chẳng qua chỉ muốn trích phần tháng trong cột ngày sinh.Msql cung cấp rất nhiều chức năng trích đẫn phần ngày sinhnhư AR(),MONTH(),DAY OF MONTH() ,MONTH() đều là chức năng phức hợp .ở đây để xem nó làm việc ra sao ,chạy một câu lệnh đơn giản để hiển thị giá trị của cả ngày sinh và tháng sinh

mysql> SELECT name, birth, MONTH(birth)
FROM pet;

Name	birth	MONTH(birth
Fluffy	1993-02-04	2

- Việc tìm các con vật có ngày sinh vào tháng sau hết sức dễ dàng .Giả sử tháng hiện tại có tháng tư .Sau đó ,giá trị tháng là 4 và bạn tìm các con vật được sinh vào tháng năm như sau:
    - mysql> SELECT name, birth FROM pet WHERE MONTH(birth) = 5;
- Tất nhiên có một rắc rối nhỏ nếu tháng hiện tại là tháng 12.Bạn không chỉ thêm 1 vào số tháng (12) và tìm xem con vật nào sinh vào tháng 13 ,bởi vì không có tháng như vậy .Thay vào đó ,bạn tìm các con vật sinh vào tháng 1
- Bạn cũng có thể viết câu lệnh mà nó làm việc không quan tâm tới tháng hiện tại là bao nhiêu ,với cách đó bạn không phải sử dụng 1 số tháng riêng biệt nào trong câu lênh DATE=AĐ() cho phép bạn thêm một khoảng thời gian vào một ngày đã được đưa ra .Nếu bạn thêm 1 tháng vào giá trị của NULL() ,sau đo trích phần tháng với MONTH(),kết quả tạo ra tháng mà chúng ta tìm ngày sinh trong đó

```
mysql> SELECT name, birth FROM pet
->WHERE MONTH(birth) =
MONTH(DATE_ADD(NOW(), INTERVAL 1
MONTH));
```

- Một cách khác để hoàn thành công việc tương tự là phải thêm 1 vào để có tháng sau tháng hiện tại (sau khi sử dụng chức năng module MOD) để "bọc xung quanh " giá trị tháng tới 0 nếu nó hiện tại là tháng 12

```
mysql> SELECT name, birth FROM pet
➔ WHERE MONTH(birth) = MOD(MONTH(NOW()), 12)  1;
```

- Chú ý rằng MONTH quay vòng là một số giữ tháng 1 và tháng 12 ,và MOD quay vòng là một số giữa 0 và 11.Do Đó sự thêm phải theo sau MOD() nếu không chúng ta sẽ đi từ tháng 11 đến tháng 1

5. Làm việc với giá trị NULL

- Giá trị NULL có thể gây ngạc nhiên cho đến khi bạn làm quen với nó .Theo khái niệm NULL có nghĩa là"giá trị không có"hoặc "giá trị không được biết đến" và nó được xem xét ở

một mức độ khác biệt hơn so với giá trị khác .Để kiểm tra NULL bạn không thể sử dụng các toán tử so sánh số học như:=,>,<...Để tự mình giải thích điều này bạn hãy thử câu lệnh sau

mysql> SELECT 1 = NULL, 1 != NULL, 1 < NULL, 1 > NULL;

1 = NULL	1 != NULL	1 < NULL	1 > NULL
NULL	NULL	NULL	NULL

- Rõ ràng là bạn không có các kết quả có nghĩa từ các sự đối chiếu này hãy sử dụng
  IS NULL và IS NOT NULL để thay thế.
  mysql> SELECT 1 IS NULL, 1 IS NOT NULL;

1 IS NULL	1 IS NOT NULL
0	1

- Trong Mysql ,0 có nghĩa là sai và 1 có nghĩa là đúng
- Cách sử lý đặc biệt này của NULL là lý do tại sao trong mục trước việc xác định con vật nào đã chết bằng việc sử dụng death =NOT NULL thay vì death!=NULL ,lại cần thiết như vậy

6. Sự phù hợp về kiểu

- Mysql cung câp các mẫu chuẩn cho phép thay thế một kiểu nào đó:ví dụ như sử dụng'_'để chỉ một ký tự đơn lẻ nào đó '%'để chỉ một số ký tự nào đó có thể là không có ký tự nào.Chú ý rằng trong mysql không sử dụng dấu=hay!= mà sử dụng LIKE hoặc NOT LIKE.
- Vi dụ tìm một tên bắt đầu bằng ký tự 'b'
  mysql> SELECT * FROM pet WHERE name LIKE "b%";

Name	Owner	species	sex	birth
Buffy	Harold	dog	F	1989-05-13
Bowser	Diane	dog	M	1989-08-31

- Để tìm tên kết thúc bằng 'fy':
  mysql> SELECT * FROM pet WHERE name LIKE "%fy";
- Để tìm tên có chứa một ký tự bằng 'w':
  mysql> SELECT * FROM pet WHERE name LIKE "%w%";
- Dùng LIKE:
  mysql> SELECT * FROM pet WHERE name LIKE "_____";

- Một số các quy tắc khác:'
  .Một lớp ký tự:được bao bởi '[...]',ví dụ' [abc]''[1-9]'
  .Ký hiệu '*'để chỉ ký hiệu trống
  .Sử dụng ký hiệu '^''$'để chỉ bắt đầu hoặc kết thúc
  để sử dụng thi ta phảI viết từ khoá REGEXP phía trước
- Tìm tên bắt đầu với ký tự 'b'

```
mysql> SELECT * FROM pet WHERE name REGEXP
"^[bB]";
```

- Tìm tên kết thúc bằng "fy",sử dụng '$' :

```
mysql> SELECT * FROM pet WHERE name REGEXP
"fy$";
```

- Tìm tên có chứa ký tự'w'sử dụng '[wW]'để tìm cả ký tự thường và ký tự hoa:

```
mysql> SELECT * FROM pet WHERE name REGEXP
"[wW]";
```

- Tìm tên có chứa đúng 5 ký tự sử dụng'^'và '$'để tìm kiếm:

```
mysql> SELECT * FROM pet WHERE name REGEXP
"^.....$";
```

Bạn cũng có thể thực hiện như sau:

```
mysql> SELECT * FROM pet WHERE name REGEXP
"^.{5}$";
```

7. Đếm hàng

- Các cơ sở dữ liệu thường được sử dụng để trả lời câu hỏi "mức độ thường xuyên của một số kiểu dữ liệu xuất hiện trong bảng như thế nào"Ví dụ ,bạn có thể muốn biết bạn có bao nhiêu con vẹt ,hoặc mỗi người chủ có bao nhiêu con vẹt ,hoặc bạn có thể thực hiện rất nhiều kiểu điều tra dân số các con vật của mình
- Đếm tổng số con vật mà bạn có thì tương tự như câu hỏi là "có bao nhiêu hàng trong bảng về con vẹt" khi đó một bản ghi trên một con vẹt ,chức năng COUNT() đếm số kết quả non-NULL,do đó câu lệnh đếm con vật của bạn như sau:

```
mysql> SELECT COUNT(*) FROM pet;
```

- Dễ dàng hơn ,bạn có thể gọi tên của những người chủ sở hữu các con vẹt.Bạn có thể sử dụng COUNT() nếu bạn muốn tìm ra mỗi người chủ có bao con vẹt

```
mysql> SELECT owner, COUNT(*) FROM pet
GROUP BY owner;
```

- Chú ý rằng ,sử dụng GROUP BY để tập hợp lại tất cả các bản ghi của mỗi người chủ.Nếu không có nó,tất cả những cái mà thu dược chỉ là 1 bức điện báo lỗi

mysql> SELECT owner, COUNT(owner) FROM pet;

ERROR 1140 at line 1: Mixing of GROUP columns (MIN(),MAX(),COUNT()...)

with no GROUP columns is illegal if there is no GROUP BY clause

- COUNT() và GROUP BY rất có ích cho việc mô tả dữ liệu của bạn theo những cách khác nhau.
- Ví dụ sau đây sẽ chỉ ra những cách khác nhau để thực hiện các điều tra dân số con vật

Số con vật theo loài:

mysql> SELECT species, COUNT(*) FROM pet GROUP BY species;

Species	COUNT(*)
Bird	2
cat	2
dog	3
hamster	1
snake	1

Số con vật theo giới tính:

mysql> SELECT sex, COUNT(*) FROM pet GROUP BY sex;

(trong phần này ,NULL ám chỉ không biết giới tính)

Sex	COUNT(*)
NULL	4
F	1
M	1

Số con vật theo cả loài và giới tính:

mysql> SELECT species, sex, COUNT(*) FROM pet GROUP BY species, sex;

- Bạn không cần thao tác trên toàn bộ một bảng khi sử dụng COUNT().Ví dụ câu lệnh trước khi được thực hiện chỉ trên chó và mèo ,như sau:

mysql> SELECT species, sex, COUNT(*) FROM pet

-> WHERE species = "dog" OR species = "cat"

```
-> GROUP BY species, sex;
```

- Hoặc nếu bạn muốn biết số con vật theo giới tính và chỉ với những con đã biết giới tính:

```
mysql> SELECT species, sex, COUNT(*) FROM pet
  -> WHERE sex IS NOT NULL
  -> GROUP BY species, sex;
```

III. Sử dụng nhiều hơn một bảng

- Bảng về con vẹt chỉ giữ thống kê là bạn có những con vẹt nào .Nếu bạn muốn ghi các thông tin khác về chúng ,như là các sự kiện trong cuộc sống của chúng như việc đến gặp bác sĩ thú y ,hoặc các lứa đẻ được sinh khi nào ,bạn cần một bảng khác ,vậy bảng này nên cần gì nó cần chứa tên con vẹt ,vì vậy bạn có thể biết mỗi sự kiện nói đến con vật nào

  Cần ngày sinh để bạn biết được sự kiện đã xảy ra khi nào

  Cần một trường để mô tả sự kiện

  Nếu bạn muốn có thể phân loại được các sự kiện thì việc có một trường kiểu sự kiện rất hữu ích

- Khi những yêu cầu đó được đưa ra việc trình bày CREATE TABLE cho bảng co thể như sau:

```
mysql> CREATE TABLE event (name
VARCHAR(20), date DATE,
->type VARCHAR(15), remark VARCHAR(255));
```

- Với bảng về con vẹt ,việc chèn các bản ghi cần tiến hành bằng việc thiết lập các vùng để lưu trữ thông tin.

Fluffy	1995-05-15	litter	4 kittens, 3 female, 1 male
Buffy	1993-06-23	litter	5 puppies, 2 female, 3 male
Buffy	1991-10-12	litter	3 puppies, 3 female
Chirpy	1995-05-15	vet	needed beak straightened
Slim	1998-08-28	vet	broken rib
Bowser	1998-12-09	kennel	
Fang	1993-06-23	kennel	
Fang	1998-12-09	birthday	Gave him a new chew toy
Claws	1998-08-28	birthday	Gave him a new flea collar

- Chèn các bản ghi như sau:

```
mysql> LOAD DATA LOCAL INFILE "event.txt"
INTO TABLE event;
```

- Dựa trên những gì mà bạn học được từ các câu lệnh chạy bảng "pet" bạn có thể thực hiện việc gọi tên trên các bản ghi trong bảng event,nguyên tắc vẫn không thay đổi .Nhưng khi bản thân bảng event không đủ để trả lời các câu hỏi mà bạn có thể hỏi ?
- Giả sử bạn muốn tìm tuổi của mỗi con vật khi nó có lứa đẻ .Bảng evevt chỉ ra điều đó xuất hiện khi nào ,nhưng để tính toán tuổi của con mẹ bạn cần ngày sinh của nó .Nó được lưa trữ trong bảng pet,nên bạn cần cả hai bảng cho câu lệnh này:

```
mysql> SELECT pet.name, (TO_DAYS(date) -
TO_DAYS(birth))/365 AS age, remark
   -> FROM pet, event
->WHERE pet.name = event.name AND type = "litter";
```

- Có rất nhiều điều đáng lưu ý về câu lệnh này

Mệnh đề FROM ghi vào danh sách cả hai bảng khi câu lệnh cần lấy thông tin từ cả hai

Khi kết hợp thông tin từ các bảng khác nhau ,bạn cần định rõ có bao nhiêu bản ghi và mỗi bản ghi có thể được nối với các bản ghi của bản kia.Điều đó rất dễ khi cả hai bản ghi đều có cùng số cột.Câu lệnh sử dụngWHERE dùng để nói các bản ghi ở hai bảng dựa trên giá trị tên

Khi tên cột xuất hiện cả ở hai bảng ,bạn phải xác định về bản nào bạn tìm đến kiểu ám chỉ cột .Điều này được thực hiện bằng việc quy chiếu tên của bảng với tên của cột

- Bạn không nhất thiết phải có hai bảng khác nhau để tiến hành liên kết .Đôi khi việc tự liên kết lại rấ hữu ích ,nếu bạn muốn

IV. Kết nối và không kết nối tới server

1. Để kết nối tới server bạn thường cung cấp cho My SQL một username, và một password. Nếu server chạy trên một máy khác hơn là máy bạn đang login. Bạn cũng cần xác định một hostname để liên lạc với người quản trị của bạn để tìm ra những kết nối mà bạn sẽ sử dụng để kết nối.

Khi bạn biết chính xác hostname,username,password bạn có thể kết nối như sau :

```
shell> mysql -h host -u user -p
Enter password: ********
```

2. Một vài My SQL ổn định cho phép người sử dụng kết nối mà không có tên từ user tới server đang chạy localhost trong những trường hợp này bạn có thể kết nối tới server đó như sau:

   shell> mysql

   Sau khi bạn bạn đã kết nối thành công bạn có thể kết thúc việc kết nối như sau :

   mysql> QUIT

   Bạn cũng có thể sử dụng Ctr-D.

VI. Ví dụ về các lệnh cơ bản

*Sau đây là những ví dụ về cách giải quyết vấn đề cùng với mysql

* Một vài ví dụ sử dụng hàng shop để lưu trữ giá của mỗi đề mục cho mỗi thương nhân.Giả sử rằng mỗi thương nhân có mộ tập hợp giá cho mỗi đề mục .Sau đó là khoá cho mỗi bản ghi

Bạn có thể tạo ra một bảng như sau:

```
CREATE TABLE shop (
 article INT(4) UNSIGNED ZEROFILL DEFAULT '0000' NOT NULL,
dealer  CHAR(20)    DEFAULT ''     NOT NULL,
 price   DOUBLE(16,2)   DEFAULT '0.00' NOT NULL,
 PRIMARY KEY(article, dealer));
INSERT INTO shop VALUES
(1,'A',3.45),(1,'B',3.99),(2,'A',10.99),(3,'B',1.45),(3,'C',1.69),
(3,'D',1.25),(4,'D',19.95);
```

*Ví dụ về dữ liệu như sau:

SELECT * FROM shop

article	Dealer	Price
0001	A	3.45
0001	B	3.99
0001	A	10.99
0002	G	1.45
0003	D	1.69
0004	F	1.25
		19.95

1. Giá trị lớn nhất cho mỗi cột

Số mục cao nhất là bao nhiêu:

```
SELECT MAX(article) AS article FROM shop
```

Article
4

2.Dòng lưu trữ giá trị lớn nhất chứa trong cột :

Tìm số dealer và price của các article đắt nhất

```
SELECT article, dealer, price
FROM   shop
WHERE  price=(SELECT MAX(price) FROM shop)
```

(trong mysql không có thủ tục lựa chọn)
nên để làm điều đó phải chia thành hai bước
Tìm giá cao nhất từ bảng bằng lệnh SELECT
Sử dụng giá trị này để hoàn thành các câu lệnh tiếp theo

```
SELECT article, dealer, price
FROM   shop
WHERE  price=19.95
```

Một cách làm khác là sắp xếp các hàng giảm dần theo giá và chỉ đưa ra hàng đầu tiên sử dụng LIMIT:

```
SELECT article, dealer, price
FROM   shop
ORDER BY price DESC
LIMIT 1
```

3. Giá trị mã của cột:mỗi nhóm:chỉ một giá trị

* với mỗi article ,tìm dealer(s) sao cho giá là cao nhất
* trong ANSI SQL nó sẽ làm cùng với thủ tục như sau

```
SELECT article, dealer, price
FROM   shop s1
WHERE  price=(SELECT MAX(s2.price)
  FROM shop s2
  WHERE s1.article = s2.article)
```

*Trong mysql tốt nhất là làm từng bước

a.Đưa ra danh sách gồm 2 cột(article,maxpice)
b.Với mỗi article chọn những dòng mà có giá cao nhất

*Ví dụ như sau:

```
CREATE TEMPORARY TABLE tmp (
      article INT(4) UNSIGNED ZEROFILL
DEFAULT '0000' NOT NULL,
      price   DOUBLE(16,2)          DEFAULT '0.00'
NOT NULL);

LOCK TABLES article read;
```

```
INSERT INTO tmp SELECT article, MAX(price)
FROM shop GROUP BY article;

SELECT article, dealer, price FROM shop, tmp
WHERE shop.article=tmp.article AND
shop.price=tmp.price;

UNLOCK TABLES;

DROP TABLE tmp;
```

*Nếu bạn không sử dụng một temporary table bạn phải dùng khoá "tmp" table
*Có thể làm điều đó với những câu lệnh đơn lẻ
*Có thể gọi hàm "MAX-CONCAT"

```
SELECT article,
      SUBSTRING( MAX(
CONCAT(LPAD(price,6,'0'),dealer) ), 7) AS dealer,
  0.00LEFT(      MAX(
CONCAT(LPAD(price,6,'0'),dealer) ), 6) AS price
FROM   shop
GROUP BY article;
```

4. Sử dụng khoá ngoài

Bạn không cần khoá ngoài để nối hai bảng

Chỉ những thứ trong mysql không làm được CHECK để chắc chắn rằng những khoá bạn sử dụng thực sự tồn tại trong bảng liên quan và nó không tự động xoá các dòng từ bảng cùng với khoá ngoài được định nghĩa.Nếu bạn sử dụng khoá bình thường nó vẫn làm việc tới cùng:

```
CREATE TABLE persons (
 id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT,
   name CHAR(60) NOT NULL,
   PRIMARY KEY (id)
);

CREATE TABLE shirts (
   id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT,
   style ENUM('t-shirt', 'polo', 'dress') NOT NULL,
   color ENUM('red', 'blue', 'orange', 'white', 'black') NOT NULL,
   owner SMALLINT UNSIGNED NOT NULL REFERENCES persons,
   PRIMARY KEY (id)
);
```

```
INSERT INTO persons VALUES (NULL, 'Antonio Paz');

INSERT INTO shirts VALUES
(NULL, 'polo', 'blue', LAST_INSERT_ID()),
(NULL, 'dress', 'white', LAST_INSERT_ID()),
(NULL, 't-shirt', 'blue', LAST_INSERT_ID());

INSERT INTO persons VALUES (NULL, 'Lilliana Angelovska');

INSERT INTO shirts VALUES
(NULL, 'dress', 'orange', LAST_INSERT_ID()),
(NULL, 'polo', 'red', LAST_INSERT_ID()),
(NULL, 'dress', 'blue', LAST_INSERT_ID()),
(NULL, 't-shirt', 'white', LAST_INSERT_ID());
SELECT * FROM persons;
```

Id	Name
1	Antonio Paz
2	Lilliana Angelovska

```
SELECT * FROM shirts;
```

Id	style	color	owner
1	Polo	blue	1
2	Dress	white	1
3	t-shirt	blue	1
4	Dress	orange	2
5	Polo	red	2
6	Dress	blue	2
7	t-shirt	white	2

```
SELECT s.* FROM persons p, shirts s
 WHERE p.name LIKE 'Lilliana%'
   AND s.owner = p.id
   AND s.color <> 'white';
```

Id	style	color	owner
4	Dress	orange	2
5	Polo	red	2
6	Dress	blue	2

CHƯƠNG IV : HỆ THỐNG QUYỀN TRUY NHẬP CƠ SỞ DỮ LIỆU TRONG MYSQL

I. Giới thiệu chung về vấn đề an toàn và bảo mật của hệ thống

Trong việc thảo luận về vấn đề an toàn của hệ thống chúng ta dặc biệt nhấn mạnhviệc cần thiết của việc bảo vệ an toàn cho máy chủ (không đơn giản là máy chủ Mysql) chống lại sự tấn công dưới mọi hình thức : nghe trộm, sửa đổi cấu trúc ….

Mysql sử dụng danh sách điều khiển truy nhập (ALSC) (ACCESS CONTROL LIST) để đảm bảo an toàn bảo mật của việc kết nối,câu truy vấn và các thao tác khác mà người sử dụng cố gắng thực hiện cùng tồn tại một số sự hổ trợ mã hoá kết nối giữa máy chủ Mysql và máy khách .

Khi dùng Mysql chúng ta cần chú ý các điều sau đây:

- không cho phép người sử dụng truy cập đến bảng Mysql (ngoại trừ những người dùng root)
- học hệ thống các quyền truy xuất của Mysql các câu lệnh cấp phát quuyền và huỷ quyền của người sử dụngnhằm mục đích ngăn chặn việc truy xuất tới Mysql, không nên cấp phát quyền hạn nhiều hơn mức cần thiết, không nên cấp quyền tới tất cả các máy chủ.

Bạn cần kiểm tra những điều sau đây:

- cố gắng thực hiện Mysql-u root ,nếu bạn có thể kết nối với máy chủ mà không cần mật khẩu, bạn đã có một số vấn đề rồi đó, bất kỳ một người nào cũng có thể kết nối với máy chủ Mysql vớiđầy đủ quyền truy xuất, bạn cần xem lại các chỉ thị khi cài đặt Mysql, chú ý đến việc đặt mật khẩu gốc (root).
- Sử dụng lệnh show grant và kiểm tra người dùng truy cập đến dẽ liệu gì, thu hồi bớt quyền hạn không cần thiết .
- Không nên lưu mật khẩu dưới dạng văn bản thuần tuý trong cơ sở dữ liệu, vì khi máy của bạn bị một kẻ khác truy nhập thì những kẻ truy nhập có thể có đầy đủ một danh sách các mật khẩu và sử dụng chúng.
- Không nên dùng mật khẩu ở từ điển vì có nhiều chương trình đặc biệt để phá mật khẩu này .
- Hãy dung filewall nó sẽ bảo vệ tối thiểu là 50% tất cả các kiểu khai thác khác nhau của phần mền vì vậy hãy đặt mysql bên cạnh firewall
- Kiểm tra người dùng có nhập dữ liệu đáng tin cậy không
- Không nên truyền tải các dữ liệu chưa được mã hoá trên mạng vì dữ liệu đó có thể bị chặn lại để sử dụng .
- Học cách sử dụng các tiện ích tepdump, string để bảo vệ

II. Làm thế nào để Mysql trở nên an toàn chống lại bọn tội phạm máy tính

khi kết nối đến cơ sở dữ liệu thông thường bạn hay dùng một mật khẩu, tuy nhiên việc mã hoá chúng thì không htực sự mạnh và một số nổ lực tấn công của những kẻ phá hoại có thể bẻ được mật khẩu của bạn nếu như những kẻ phá hoại có thể chặng đứng được việc lưu thông giữa máy chủ và máy khách, tất cả các thông tin truyền trên mạng đều có thể bị người nào đó chặng đứng lại và sử dụng chúng, nếu bạn lo lắng về điều này thì bạn có thể sử dụng việc mã hoá kết nối dạng tep/ip khi kết nói giữa máy chủ và máy khách.

Để đảm bảo an toàn cho hệ thống bạn cần thực hiện những việc sau :

- Sử dụng password cho tất cả người dùng vì nếu không dùng password thì bất kỳ người dùng nào cũng có thể truy nhập cơ sở dữ liệu
- Bạn có thể thay đổi password cho tất cả người dùng bằng chách thay đổi kịch bản Mysql_install_db trước khi thực hiện mysql hoặc sửa password cho người dùng root
- sell>mysql-u root mysql;
  mysql>update user set password =password (‘neư password’);
  where user =’root’;
  mysql>flúh privileges;
- Không nên chạy mysql như người dùng unix root điều này rất nguy hiểm vì người có quyền đói với file có thể tạo file tương tự như root, để ngăn chặn điều này thì mysql thường không chạy mysql như dạng root trừ khi có chỉ thị trực tiếp là :
  -user=root;
  Nếu đặt mật khẩu cho người dùng unix root trong kịch bản của mysql server thì bạn phải đảm bảo rằng nó chỉ có thể được đọc bởi người dùng root.
- Kiểm tra người dùng Unix rằng mysql chỉ chạy đối với người dùng có quyền read/write.
- Trong các trạm làm việc Unix không nên chạy Mysql như người dùng root trừ phi điều đó thực sự cần thiết .
- Không nên gán quyền process cho tất cả người dùng .
- Không nên gán quyền đôío với file cho tất cả người dùng .

Một số sự chọn lựa liên quan đến việc an toàn khi khởi động Mysql

-- secure : trả lại số ip bởi gethost by name() lời gọi hệ thống sẽ kiểm tra để đảm bảo chắc chắn rằng chúng sẽ trả lại tên của máy chủ, điều này sẽ gây khó khăn cho những ai phịa ra một tên máy để lấy địa chỉ.

-- skip-grant-table : điều này sẽ làm cho hệ thống không sử dụng hệ thống quuyền và tất cả mọi người đều có đầy đủ quyền hạn để truy nnhập vào cơ sở dữ liệu (bạn có thể sử dụng hệ thống quyền bằng cách thực hiện mysqladmin flush-privileges).
-- skip-name-resolve: sự chọn lựa này sẽ làm cho tất cả các giá trị trong cột host của bảng phân quyền đều là địa chỉ IP hoặc là localhost.
-- skip-networking: không cho phép giao thức tep/ip được kết nối .
-- skip-show database: khi có lệnh SHOW DATABASE sẽ không trả lại một giá trị nào cả .
-- safe-show-database: lệnh SHOW DATABASE sẽ chỉ trả lại giá trị đối với những người có quyền đối với lệnh này .

III. Tên người dùng và mật khẩu trong mysql

Có rất nhiều nét phân biệt giữa các cách thức khác nhau mà tên người dùng (user name) và mật khẩu(pasword) được sử dụng trong Mysql và cách thức chúng được dùng trong Windows và Unix.
User nameđược mysql sử dụng cho mục đích nhận dạng hầu hết máy khách Myssql mặc định là khi đăng nhập vào hệ thống sử dụng tên người dùng Unix hiện tại điều này có một điều không hay. chương trình ở máy khách cho phép nhiều tên người dùng khác nhau được chỉ ra với lựa chọn –u hoặc –user điều này sẽ làm cho hệ thống trở nên không an toàn trong mọi trường hợp trừ khi tất cả các tên người dùng đều co password bổ vì bất kỳ một người nào đó cũng có thể nối với máy chủ sử dụng một tên nào đó và họ sẽ thành công nếu như tên đó không có password .

- Tên của người dùng có thể dài 16 kí tự.
- Password trong Mysql không có liên hệ đối với password trong Unix. Không có sự liên hệ cần thiết giữa mật khẩu bạn sử dụng để đănh nhập vào máy tính unixvowis mật khẩu bạn dùng để đăng nhậo vào cơ sở dữ liệu

Giữ mật khẩu một cách an toàn

Một diều không hay nếu như mật khẩu của bạn bị một người khác phát hiện ra.Dưới đây là những phương pháp mà bạn chỉ rõ mật khẩu của mình và các đánh giá về các nguy hiểm cho từng phương pháp .
Sử dụng –pyour_pass hoặc –password=your_pass trên dòng lệnh, điều này thì tiện lợi nhưng không an toàn vì mật khẩu cảu bạn sẽ rõ ràng đối với chương trình trạng thái làm việc của hệ thống, và một
người dùng khác có thể thực hiện một lệnh hiển thị.
Sử dụng –p trong trường hợp này chương trình ở máy khách sẽ yêu cầu nhập mật khẩu từ dòng lệnh

```
Sell>mysql-u user_name-p
```

Enter password :********

Việc nhập mật khẩu sẽ an toàn hơn vì nó sẽ không bị nhìn thấy bởi người dùng khác. Tuy nhiên việc nhập mật khẩu kiểu này chỉ thích hợp cho những chương trình mà bạn chạy theop lối tương tác .Nếu như bạn gọi một kịch bản của máy khách thì sẽ không có cơ hội để nhập mật khẩu từ thiết bị đầu cuối .

Bạn cũng có thể dùng phương pháp lưu mật khẩu của bạn vào file cấu hình (ví dụ bạn có thể tạo ra một danh sách các mật khẩu trong phần [client] của file 'My.cnf' trong Home Directory của mình nếu theo cách này file không nên để dạng có thể đọc ghi.

Bạn có thể lưu mật khẩu trong biến môi trường MySQL_pwd nhưng trường hợp này thường ít dùng vì nó không an toàn.

IV. Hệ thống quyền truy xuất được cung cấp bởi MySQL

Thông tin về quyền truy xuất của người sử dụng được lưu trữ trong các bảng User, D, Host, Table_Priv và bảng Column_Priv trong cơ sở dữ liệu MySQL, MySQL server sẽ đọc nôit dung của các bảng khi khởi động.

Bảng sau đây liệt kê các quyền và tương ứng với các cột liên quan đến quyền và ngữ cảnh mà các quyền đó áp dụng.

Các quyền	Cột	Ngữ cảnh
Select	Select_Priv	Bảng
Insert	Insert_Priv	Bảng
Update	Update_Priv	Bảng
Delete	Delete_Priv	Bảng
Index	Index_Priv	Bảng
Alter	Alter_Priv	Bảng
Creat	Creat_Priv	CSDL, Bảng, Chỉ số
Drop	Drop_Priv	CSDL hoặc Bảng
Grant	Grant_Priv	CSDL hoặc Bảng
References	References_Priv	CSDL hoặc Bảng
Reload	Reload_Priv	Quản trị máy chủ
Shutdown	Shutdown_Priv	Quản trị máy chủ
Process	Process_Priv	Quản trị máy chủ
file	file_Priv	Quản trị máy chủ

- Quyền Select, Insert, Delete, Update cho phép bạn thực hiện các thao tác đối với các hảng của bảng trong cơ sở dữ liệu.
- Quyền Alter cho phép bạn sử dụng câu lệnh Alter Table.
- Quyền Creat, Drop cho phép bạn tạo mới một CSDL, bảng hoặc xoá CSDL bảng đã tồn tại, chú ý rằng nếu bạn cấp phát quyền Creat, Drop cho người sử dụng, thì người dùng có thể xoá CSDL mà trong đó quyền truy nhập đến MySQL được lưu trữ.

- Quyền Grant cho phép bạn cấp phát quyền cho những người dùng khác những quyền mà bạn có.
- Quyền đối với file cho phép đọc với ghi đối với file trên server, sử dụng các lệnh LOAD DATA INFILE và SELECT.. INTO OUTFILE bất kì người dùng nào được gán quyền này đều có thể đọc hoặc hoặc ghi đối với file mà SQL có thể đọc hoặc ghi.
- Những quyền còn lại được phục vụ cho công việc quản trị mà chúng được thực hiện bởi chương trình MySQL Admin. Bảng dưới đây sẽ chỉ cho bạn những lệnh mà quyền quản trị cho phép bạn thực hiện.

Quyền	***Những lệnh có thể thực hiện***
Reload	Reload, refresh, flush_privileges, flush-hosts, flush-logs, flush-tables
Shutdown	Shutdown
Process	Processlist, kills

Lệnh Reload sẽ chỉ thị cho máy chủ thực hiện việc đọc lại bảng các phân quyền, lệnh refresh nạp lại tất cả các bảng, mở và đóng tất cả các file log. Lệnh flush-privileges tương tự như lệnh reload, một số lệnh flush-* khác thực hiện chức năng như refresh, nhưng có giới hạn hẹp hơn lệnh refresh, nó sẽ thích hợp trong một số trường hợp, ví dụ ban chỉ càn Flush đối với log file thì lệnh flush-logs thích hợp hơn lệnh refresh.

- Lệnh Shutdown sẽ làm cho máy chủ ngừng hoạt động.
- Processlist sẽ hiển thị những thông tin trong suốt quá trình hoạt động với máy chủ.
- Lệnh kills sẽ tắt một phiên làm việc với máy chủ, bạn có thể thực hiện 2 lệnh trên nhiều lần trong một dòng làm việc nhưng bạn phải có quyền Process.

Một ý kiến hay tổng quát trong việc gán quyền cho người dùng đó là chỉ gán những quyền cần thiết cho người dùng mà họ cần để họ thực hiện những công việc của mình.

Một số chú ý đối với đặc điểm thực hiện những quyền sau:

- Quyền Grant cho phép người dùng cấp những mà họ có cho người dùng khác. Hai người sử dụng có các quyền khác nhau, với quyền Grant họ có thể kết hợp những quyền này.
- Quyền Alter có thể được sử dụng để phá vỡ những quyền bằng việc đổi tên bảng.
- Quyền đối với file có thể bị lạm dụng để đọc một số file trên máy chủ vào bảng trong cơ sở dữ liệu, nội dung của bảng này có thể bị truy nhập bằng cách sử dụng những câu lệnh Select, điều này cũng có thể được thực hiện đối với một số cơ sở dữ liệu chính của máy chủ.

- Quyền Shutdown có thể lạm dụng để từ chối một số dịch vụ đối với người sử dụng khác bằng cách tắt máy chủ.
- Quyền process có thể được sử dụng để xem dạng văn bản thuần Text của những câu truy vấn hiện thời kể cả những câu truy vấn để đặt lại mật khẩu
- Quyền đối với những cơ sở dữ liệu mysql có thể bị sử dụng để thay đổi password và một số quyền truy nhập thông tin khác,

Một số vấn đề mà bạn không thể thực hiện đối với hệ thống quuyền hạn trong Mysql

Bạn không thể chỉ ra một cách rõ ràng rằng người sử dụng không thể thực hiện được, nghĩa là bạn không thể xác định một người dùng phù hợp và từ chối việc kết nối .

Bạn không thể chỉ rõ một người dùng có quyền tạo và xoá các bảng trong một cơ sỏ dữ liệu, nhưng không có quyền tạo và xoá chính CSDL đó.

V. Hệ thống quyền hoạt động như thế nào

Hệ thống quyền trong Mysql đảm bảo một cách chắc chắn rằng mọi người dùng chỉ được thực hiện những công việc mà họ được cho phép thực hiện. khi bạn kết nối với máy chủ Mysql thì những đặc tính của bạn sẽ được xác định bởi máy chủ từ chính nơi mà bạn kết nối và tên người dùng mà bạn nhập. hệ thống quyền sẽ theo những đặc tính của bạn và những gì bạn muốn để thực hiện

Mysql xem xét cả hostname và username được bạn chỉ ra vì một số lí do đơn giản là tên cấp cho người dùng cung cấp sẽ thuộc về người ở mọi nơi trên mạng. ví dụ người dùng có tên là Bill kết nối vào Whitehouse.gov không cần thiết phải giống ngươiừ dùng Bill kết nối vào micrsoft.com. Mysql điều khiển điều này bằng cách cho phép bạn phân biệt nhiều người dùng trong những máy chủ khác nhau mà có cùng tên giống nhau, bạn có thể gán cho người dùng có tên là Bill một tập hợp các quyền để kết nối vào Whitehouse.và một tập hợp các quyền khác để kết nối vào micrsoft.com

Mysql điều khiển truy nhập bao gồm hai khung cảnh khác nhau

Khung cảnh 1: máy chủ sẽ kiểm tra bạn có quyền kết nối hay không

Khung cảnh 2: khi bạn kết nối với máy chủ thì Mysql sẽ kiểm tra những yêu cầu mà bạn đưa ra để xem bạn có đủ quyền hạn để thực hiện nó một cách thích đáng. ví dụ nếu bạn đã kết nối vào CSDL và cố gắng thực hiện một lệnh Select hoặc xoá một bảng của CSDL thì Mysql sẽ kiểm tra xem bạn có quyền select hoặc drop để thực hiện điều đó hay không .

Mysql dùng các bảng USER,DB,HOST trong CSDL mysql để điều khiển truy nhập ở cả 2 khung cảnh trên, các trương trong các bảng này được cung cấp dưới đây:

Tên bảng	USER	DB	HOST
Các trường phạm vi	Host	Host	Host
	User	Data	DB
	Password	User	
Các trường chỉ quyền	Select_Priv	Select_Priv	Select_Priv
	Insert_Priv	Insert_Priv	Insert_Priv
	Update_Priv	Update_Priv	Update_Priv
	Delete_Priv	Delete_Priv	Delete_Priv
	Index_Priv	Index_Priv	Index_Priv
	Alter_Priv	Alter_Priv	Alter_Priv
	Creat_Priv	Creat_Priv	Creat_Priv
	Drop_Priv	Drop_Priv	Drop_Priv
	Grant_Priv	Grant_Priv	Grant_Priv
	References_Priv		
	Reload_Priv		
	Shutdown_Priv		
	Process_Priv		
	file_Priv		

Mỗi bảng phân quyền bao gồm trường phạm vi và trường quyền, trường phạm vi xác định các phạm vi các mục trong bảng các khung cảnh là nơi các mụ đó khép vào. ví dụ trong bảng người dung mục HOST ,USER có giá trị
Là ‘vnuh.vnn.vn’và ‘thu’ dùng để nhập ra sự kết nối tới máy chủ bởi ‘thu’kết nối tới máy chủ ‘vnuh.vnn.vn’tương tự trong bảng DB mục HOST ,USER, DB có các giá trị là ‘vnuh.vnn.vn’và ‘thu’, ‘repost ’ dùng để nhận ra sự kết nối bởi ‘thu’ kết nối tới máy chủ ‘vnuh.vnn.vn’, truy nhập đến CSDL ‘repost’ bảng table-priv,column-priv chứa trường phạm vi để chỉ ra các bảng, bảng / cột được kết hợp lại cho các mục được yêu cầu.
-Để thực hiện chức năng kiểm tra truy nhập việc so sánh giá trị của Host thì không phân biệt chữ hoa chữ thường, giá trị của db, password và table_name thì có phân biệt chữ hoa chữ thường, column_name không phân biệt chữ hoa chữ thường.
-Trường trong quyền mỗi bảng chỉ ra các quyền được gán bởi các thực thể bảng đó là các thao tác có thể cho phép thực hiện. Trường phạm vi có kiểu xâu được khai báo như trước đây, giá trị mặc định cho mỗi trường là rỗng.

Tên trường	Kiểu trường
Host	CHAR(60)
User	CHAR(16)
Password	CHAR(16)
Db	CHAR(64)
Table_name	CHAR(60)
Column_name	CHAR(60)

Trong tất cả các bảng uer, host, db tất cả các trường đều khai báo là Enum('N','Y'), giá trị mặc định cho mỗi trường là 'N'
-Trong các bảng table_priv, column_priv các trường đều được gán kiểu tập hợp.
Đặc điểm về quyền trong các bảng của CSDL Mysql
-Bảng user:
Trường phạm vi trong bảng user xác định xem bạn có được phép thực hiện hoặc không được phép thực hiện kết nối bất kỳ một quyền nào được gán trong bảng user thì tất cả các quyền nào là toàn cục trong mysql, nghĩa là các quyền này sẽ có hiệu lực đối với tất cả CSDL trên server
-Bảng db và bảng host được sử dụng kết hợp với nhau:
Trường phạm vi trong bảng db xác định những người dùng nào được phép truy cập đến CSDL nào, máy chủ nào, trường quyền sẽ chỉ ra các thao tác nào sẽ được cho phép thực hiện.
Bảng host được sử dụng như một sự mở rộng của bảng db khi bạn muốn trao cho các thực thể trong bảng db có quyền đối với nhiều host. Ví dụ, bạn muốn một người dùng trong bảng db có thể sử dụng một CSDL từ nhiều máy chủ khác nhau.
Bảng table_priv, column_priv cũng tương tự như bảng db nhưng nó có quản lý một cách chi tiết hơn, các quyền sẽ đươc gán tại nhiều mức khác nhau cho các bảng, cột trong CSDL.
Chú ý: Các quyền phục vụ cho công tác quản trị chỉ được chỉ ra trong bảng user bởi vì một lý do đơn giản là các quyền này chỉ thao tác với máy chủ nên không có lý do gì đưa nó vào trong bảng phân quyền khác.

VI. Điều khiển truy nhập

1. Điều khiển truy nhập

Khung cảnh một (Stage 1) thẩm tra kết nối thuận hoặc từ chối việc kết nối của bạn dựa trên các đặc tính của bạn và thẩm tra lại điều này bằng cách bạn phải cung cấp một mật khẩu dúng. Nếu không nó sẽ từ chối việc kết nối của bạn, nếu thành công nó sẽ chuyển sang khung cảnh thứ hai (Stage 2).
Việc xác định những đặc tính của bạn dựa trên hai thông tin cơ bản:

- Tên máy chủ mà bạn muốn kết nối
- Tên người dùng

Việc kiểm tra sẽ thực hiện trên ba trường phạm vi trong bảng user đó là host, user và password máy chủ chi chấp nhận việc kết nối của bạn khi mà các thực thể trong bảng user phù hợp với host, user và bạn phải cung cấp password đúng.

2. Điều khiển truy nhập

Khung cảnh 2(Stage 2) xác minh yêu cầu khi việc kết nối của bạn đã thành công mysql server sẽ chuyển sang khung cảnh 2, đối với mỗi yêu cầu của mỗi kết nối mysql server sẽ thẩm tra xem bạn có đủ quyền hạn để thực hiện các quyền hạn đó hay không dựa trên các loại thao tác mà bạn muốn thực hiện việc thẩm tra sẽ đượoc thực hiện đối với các trường quyền trong các bảng phân quyền, user, host, db. Tất cả các quyền đượoc cấp trong bảng user sẽ đượoc ấn định là toàn cục cho tất cả các CSDL nói cách khác tất cả các quyền đước cấp trong bảng user là superuser.

Vì vậy mà bạn chỉ nên cấp quyền trong bảng user cho những người dùng như người quản trị máy chủ hoặc người quản trị CSDL.

VII. Khi nào việc thay đổi quyền hạn có hiệu lực

Khi mysql khởi động thì nội dung của tất cả các bảng phân quyền đượoc đặt vào bộ nhớ và khi đó nó sẽ có hiệu lực.

Việc bạn thay đổi tất cả các bảng phân quyền sử dụng lệnh GRANT (Revoke) hoặc đặt lại mật khẩu sẽ đượoc máy chủ chú ý ngay lập tức.

Nếu bạn thực hiện việc thay đổi các bảng phân quyền một cách thủ công như là sử dụng lệnh INSERT, UPDATE,… thì bạn nên dùng lệnh flush priviliges hoặc chạy mysqladmin flush priviliges hoặc mysqladmin reload để thông báo cho server biết để reload các bảng phân quyền.

Khi server thông báo một chú ý rằng các bảng phân quyền đã bị thay đổi thì các kết nối đang tồn tại ở phía client sẽ bị thay đổi như sau:

-Các thay đổi đối với các bảng, cột sẽ có hiệu lực đối với các yêu cầu tiếp theo ở phía client.

-Các thay đổi về quyền đối với CSDL sẽ có hiệu lực đối với câu lệnh userdb_name tiếp theo.

đối với các quyền toàn cục việc thay đổi password sẽ có hiệu lực ở lần kết nối tiếp theo.

VIII. Cài đặt việc khởi tạo hệ thống quyền trong mysql

-Sau khi cài đặt mysql bạn có thể đặt việc khởi tạo các quyền cho mysql bằng việc chạy scripts/mysql_install_db. Kịch bản này sẽ chạy mysqlserver để khởi các quyền bao gồm một tập hợp các quyền sau:
Đối với người dùng root sẽ được tạo như một superuser nên nó có thể làm mọi thứ, việc kết nối phải được thực hiện ở localhost.
Chú ý: Ban đầu việc khởi tạo sẽ không yêu cầu mật khẩu đối với người dùng root vì vậy bất cứ người nào cũng có thể kết nối như một người dùng root mà không bị đòi hỏi password do đó việc đầu tiêm mà bạn nên làm là thay đổi password cho người dùng root.

- Bất cứ một người dùng nào được tạo cũng có thể làm mọi thức đối với CSDL test nhưng việc kết nối phải được thực hiện ở localhost.

IX. Gán thêm quyền người dùng đối với mysql

Bạn có thể tạo một người dùng mới bằng hai cách: Dùng lệnh Grant hoặc thao tác trực tiếp trên CSDL mysql. Cách thức thường được dùng hơn cả là dùng lệnh grant bởi vì nó ngắn gọn và ít lỗi hơn.
Các ví dụ sau sẽ thực hiện việc tạo một người dùng mới. Để thực hiện điều này thì bạn phải kết nối với mysql như một người dùng root và người dùng root này phải được gán quyền insert đối với CSDL mysql và phải có quyền quản trị reload

```
Mysql> GRANT ALL PRIVILEGES ON *.* TO monty@localhost
IDENTIFIED BY ‘some_pass’ WITH GRANT OPTION;
Mysql> GRANT ALL PRIVILEGES ON *.* TO monty@”%”
IDENTIFIED BY ‘some_pass’ WITH GRANT OPTION;
Mysql> GRANT RELOAD, PROCESS ON *.* TO admin@localhost;
Mysql> GRANT USAGE ON *.* TO dummy@localhost;
```

Các câu lệnh trên gán quyền cho ban người dùng monty, admin và dummy.

- người dùng monty: Đây là người dùng superuser với đầy đủ các quyền có thể kết nối bất cứ nơi đâu nhưng phải có password là “some_pass”.
- Người dùng admin: người dùng này được quyền reload, process nhưng phải đăng nhập từ localhost và không yêu cầu password
- Người dùng dummy có thể kết nối mà không cần password nhưng không được gán quyền gì cả.
- Cũng thực hiện việc gán quyền như trên theo cách khác ta có thể làm như sau:

```
Mysql> INSERT INTO user
VALUES(‘localhost’,’monty’,PASSWORD(‘some_pass’),
‘Y’, ‘Y’, ‘Y’, ‘Y’, ‘Y’, ‘Y’, ‘Y’, ‘Y’, ‘Y’, ‘Y’, ‘Y’, ‘Y’, ‘Y’, ‘Y’)
```

```
Mysql> INSERT INTO user
VALUES('%','monty',PASSWORD('some_pass'),
'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y')
Mysql> INSERT INTO user SET Host='localhost', User='admin',
Reload_priv='Y', Process_priv='Y';
Mysql> INSERT INTO user (Host, User, Password)
VALUES('localhost','dummy','');
Mysql> FLUSH PRIVILEGES;
```