用PHP实现异构分布式数据库访问法

冯祖洪 （银川西北第二民族学院网络计算中心 750021）

摘要：本文介绍了用PHP程序实现跨平台异构分布式数据库访问的编程方法和程序例子，用这种方法，我们能方便地跨服务器交叉访问分布在多台服务器上的数据库。

关键词：数据库　分布式数据库　PHP

1 引言

笔者经过探索和实验，成功地用PHP实现了跨平台异构分布式数据库访问，在一定程度上实现了跨数据库服务器的数据协同处理，使用户能方便、有效地使用分散在各个数据库服务器上的数据。这种方法在我院的WEB服务器（系统平台是Red Hat Linux 7.1）和二台Mysql（系统平台是Red Hat Linux 7.1）、一台MS SQL server（系统平台是Windows 2000）、一台Oracle（系统平台是Windows 2000）数据库服务器之间得到了良好的实际应用。

2 用PHP程序实现跨平台异构分布式数据库访问的过程

用PHP程序实现跨平台异构分布式数据库访问的过程如图1。

如图1所示，为了实现跨平台异构分布式数据库访问，利用PHP脚本语言的平台无关特性，在WEB服务器端开发一段通用PHP脚本程序，作为一个独立的模块（图中名为select1.php）存放在WEB服务器上，协调处理跨数据库服务器的数据，实现异构分布式数据库访问，下文将给出实际应用的例子和代码。

用户用浏览器通过WEB服务器上的PHP脚本语言程序访问各数据库。WEB服务器的运行平台可以是Windows NT/2000、UNIX、Linux等支持PHP脚本语言的任何系统平台。数据库服务器的运行平台可与WEB服务器的运行平台相同，也可以不同。为了使PHP脚本语言能与各数据库服务器建立连接，运行WEB服务器的计算机上需安装各数据库管理系统对应的客户端软件。例如，只有在运行WEB服务器的计算机上安装Oracle数据库的客户端软件，PHP脚本语言程序才能与Oracle数据库服务器建立连接。值得指出的是，当WEB服务器的运行平台为UNIX或Linux时，需安装Sybase的客户端软件，PHP脚本语言程序才能与MS SQL Server数据库服务器建立连接。

当用户的访问只涉及单台数据库服务器上的数据时，PHP脚本程序按通常方法与数据库服务器建立连接并访问数据库；当用户的访问同时交叉用到多台数据库服务器上的数据时，PHP脚本程序先分别与数据库服务器建立连接，再调用select1.php访问数据库，利用select1.php，能象访问一个数据库中的多个表一样，方便地交叉访问分布在多台服务器上的多张表。例如，有二张表，表的结构见表1、表2：

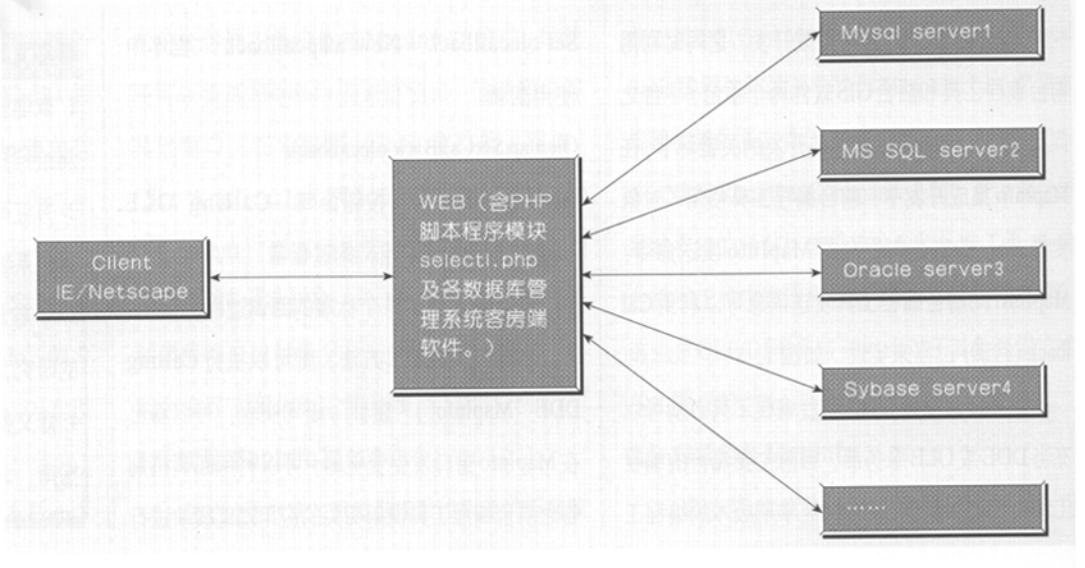
表1 xs1在图书馆的数据库服务器server1上，表2 xs2在教务处的数据库服务器server2上，PHP脚本程序中用类似"select * from xs1, xs2 where id1=id2"的语句，就能跨数据库服务器得到每个学生的成绩和借书记录，用类似"select * from xs1,xs2 where id1=id2 and sx>=90"的语句，得到高数成绩在90分以上学生的借书记录。

3 Select1.php的工作原理

Select1.php是一个PHP脚本语言模块，存放在WEB服务器上，其中的主要代码是函数select1（$db1, $db2,$str1,$str2,$var1,$var2,$str3, &$link1,&$link2,$it,$iu)。Select1()函数中各个参数的意义如下：

$db1和$db2是二个字符串型参数，可以表达为"db-name"，用于向Select1()声明$link1、$link2二个标识连接的数据库系统。

图1 PHP程序实现跨平台异构分布式数据库访问过程示意图



$str1 是一个字符串型参数，可以表达为 select-expression from table-references″,表示从二台数据库服务器的一台中应取出的内容。

$str2 与 $str1 一样，表示从另一台数据库服务器中应取出的内容。

参数$var1和$var2可以表达为″tbl-name.col-name″,用于向函数传入二个数据库表中的某个列名。这里，二个列分别属于二个数据库,对应于select语句中的″where tbl-name1.col-name1= tbl-name2.col-name2″ 部分。

参数$str3用于实现 ″where id1=id2 and sx>=90″ 语句的″ sx>=90″ 部分，它的取值有二种情况，第一种情况取值空字符串″″，对应用于实现(二)中类似″ select * from xs1,xs2 where id1=id2″ 的语句；第二种情况取值为 ″tbl-name.col-name compare-operator constant″ 的表达式，对应用于实现(二)中 ″select * from xs1,xs2 where id1=id2 and s x>=90″ 类似的语句。当$str3的取值为表达式 ″tbl-name.col-name compare-operator con stant″ 进时，为了使程序简单起见，将表达式″ tbl-name.col-name″ 缺省定为与$str2中的引用表属同一数据库,这需在调用函数时充分注意：″compare-operator″可以取″==″、″<=″、″>=″、″!=″、″<″、″>″ 中的任意一个；Constant取与 ″tbl-name.col-name″ 类型相同的数据常量。

参数$link1、$link2，用于向函数传入已建立的二个连接标识。这里，要注意函数中这二个参数的写法。

最后二个参数$it、$iu，是二个数值型参数，用于控制返回满足条件的记录数。当$it、$iu的取值都小于或等于零时，Select1()函数返回满足条件的所有记录。

Select1()函数的工作原理和步骤如下：

(1)函数运行后，首先由

```
if (substr($str1,0,1)==' *'){
$sql1=" select ".$str1." order by $var1";
}else{
$sql1=" select ".$var1.",".$str1." order by $var1";
}
switch ($db1){
case ('mysql'):
$result1= mysql-query($sql1,$link1);
break;
case ('mssql'):
$result1=mssql-query($sql1,$link1);
break;
case ('sybase'):
$result1=sybase-query($sql1,$link1);
break;
case ('oracle8'):
$result1= OCIParse($link1,$sql1);
break;
}
```

语句，从与连接标识$link1相关的数据库服务器上，根据$var1的值按序取出有关记录，形成记录集$result1，存放在WEB服务器的缓冲区中。定义标记变量$bj=1备用。

(2)判别参数$str3和$iu的取值，分4种情况进行处理。

①当$str3取值为空字符串佑，$iu取值为零时，由

```
if (substr($str2,0,1)==' *'){
$sql2=" select ".$str2." order by $var2 ";
}else{
$sql2=" select ".$var2.",".$str2." order by $var2
";
}
```

语句，形成标准的″select″语句字符串$sql2。

②当$str3取值为空字符串佑，$iu取值大于零时，由

```
if (substr($str2,0,1)==' *'){
$sql2=" select ".$str2." order by $var2 Limit $it,
$iu ";
}else{
```

表1 xs1

id1	int(6)	;学号
name	char(8)	;学生姓名
sm	varchar(50)	;书名
js	date	;借书时间
hs	date	;还书时间

表2 xs2

id2	int(6)	;学号
name	char(8)	;学生姓名
sx	int(3)	;高数成绩
wl	int(3)	;普物成绩
wy	int(3)	;外语成绩

表3 tfxh1

id	int(6)	;学号
name	char(8)	;学生姓名
sm	varchar(50)	;书名
js	date	;借书时间
hs	date	;还书时间

表4 tfzh2

id	int(6)	;学号
name	char(8)	;学生姓名
sx	int(3)	;高数成绩
wl	int(3)	;普物成绩
wy	int(3)	;外语成绩

TO visit database of asynchlonous distributing by PHP

```
$sql2=" select  ".$var2."," .$str2."
order by $var2 Limit $it,$iu";
}
```

语句，形成标准的"select"语句字符串$sql2。

③ 当$str3取值不为空字符串""，$iu取值为零时，由

```
if (substr($str2,0,1)==' *'){
$sql2="select ".$str2." and ".$str3."
order by $var2  ";
}else{
$sql2="select ".$var2."," .$str2." and
".$str3." order by $var2  ";
}
```

语句，形成标准的"select"语句字符串$sql2。

④ 当$str3取值不为空字符串""，$iu取值大于零时，由

```
if (substr($str2,0,1)==' *'){
$sql2="select ".$str2. " and ".$str3."
order by $var2 Limit $it,$iu  ";
}else{
$sql2="  select  ".$var2."," .$str2.
" and ".$str3." order by $var2 Limit
$it,$iu";
}
```

语句，形成标准的"select"语句字符串$sql2。

然后执行：

```
switch ($db1){
case ('mysql'):
$result2= mysql-query($sql2,$link2);
break;
case ('mssql'):
$result2=mssql-query($sql2,$link2);
break;
case ('sybase'):
$result2=sybase-query($sql2,$link2);
break;
case ('oracle8'):
$result2= OCIParse($link2,$sql2);
break;
}
```

从与连接标识$link2相关的数据库服务器上，根据$var2的值按序取出有关记录，形成记录集$result2，存放在WEB服务器的缓冲区中。

（3）如果$bj等于1，则从$result1、集$result2二个记录集中的第一条记录开始，进行交叉比较；否则，从$result1的当前记录位置，$result2中的第一条记录开始，进行交叉比较，将满足条件"tbl-name1.col-name1= tbl-name2.col-name2"的记录顺序存放到一个二维数组$af1中，直至下述情况之一发生：① 遇到记录集$result1的结束标记；② 遇到记录集$result2的结束标记；③ 二维数组$af1中的记录数等于$iu>0。如果遇到的是①、③二种情况，则转到第5步；否则，记住$result1的当前记录位置，并为标记变量$bj赋值2，转到下一步。

(4) 调整$it的值至$it+$iu,转到第2步。

（5）释放记录集$result1和$result2，返回数组$af1。

请注意：函数select1()的工作过程中，将$result1按$var1（值为某个字段名）、$result2按$var2（值为某个字段名）对记录进行了排序，如果二个记录集对应于各自的排序字段都有重复值，则在进行"tbl-name1.col-name1= tbl-name2.col-name2"比较时会产生二意性。因而，二个记录集中应至少有一个对应于自己的排序字段没有重复值记录。为了编程序方便起见，总是要求$result2在这个字段上没有重复值的记录，而对$result1，没有任何约束要求。

4 Select1.php的调用方法

调用select1.php的方法如下，设有二张表，表的结构见表3、表4：

tfzh-1位于数据库服务器fserver1的数据库f-1中，采用Mysql数据库管理系统，tfzh-2位于数据库服务器fserver2的数据库f-2中，采用MS SQL Server数据库管理系统。访问数据库的PHP脚本代码如下：

```
<?php
/* 与Mysql建立连接 */
$link1=mysql-connect("fserver1","
fzh1","fzh310");
if (!$link1)
{ echo  "database connect fail!";
exit();
}
mysql-select-db("f-1",$link1);
/* 与MS SQL Server建立连接 */
$link2=mssql-connect("fserver2","
fzh2","fzh320");
if (!$link2)
{ echo  "database2 connect fail!";
exit();
}
mssql-select-db("f-2",$link2);
/* 在这里插入select1.php */
require ("select1.php");
/* 下面语句可用标准select语句表达为：" select * from tfzh-1,tfzh-2
where tfzh-1.i d= tfzh-2.id and sx>=80
limit 1,20"，结果返回到二维数组$gh中 */
$gh=select1("mysql","mssql","*
from tfzh-1","* from tfzh-2","id","
id","sx>=80",$link1,$link2,1,20);
foreach($gh as $gj){
foreach($gj as $gi){
echo $gi." \n";
}
echo "<br>\n";
}
?>
```

5 结束语

文章介绍了跨服务器访问分布在多台服务器上数据库表的基本方法和实例程序，供有兴趣的读者参考，希望数据库的初学者能从中受益。

参考文献

1 薛军超等编，Mysql网络数据库开发，人民邮电出版社，2001年4月第1版。

2 罗明、檀文钊编著，跨平台的PHP+Mysqi，清华大学出版社，2001年11月第1版。

3 http://www.php.net