

Conexión a SQL en alojamiento compartido con PHP

Introducción

El alojamiento compartido dispone de bases de datos en MySQL y PostgreSQL, así como del servicio SQL Privado. Esta guía da varios ejemplos de cómo realizar la conexión a los distintos tipos de base de datos utilizando el lenguaje PHP.

El lenguaje PHP permite una interfaz muy sencilla con un gran número de bases de datos, incluyendo MySQL y PostgreSQL.

La comunicación con las bases de datos se hace con la ayuda de peticiones SQL, un lenguaje de 4a generación (4GL) reconocido por todo el conjunto de los SGBD (sistemas gestores de bases de datos).

Las funciones de base

El lenguaje Php provee un gran surtido de funciones que permiten manipular las bases de datos. No obstante, estas cuatro funciones son esenciales:

- La función de conexión al servidor
- La función de selección de la base de datos
- La función de consulta
- La función de desconexión

Con el SGBD MySQL estas funciones son las siguientes:

```
mysql_connect  
mysql_select_db  
mysql_query  
mysql_close
```

Con el SGBD PostgreSQL estas funciones son las siguientes:

```
pg_connect  
pg_query
```

```
pg_close
```

Probar la conexión en línea

Para evitar problemas de contraseñas, así como de parámetros, es recomendable probar la conexión a la base de datos previamente, a través de los gestores de bases de datos en línea.

Para bases de datos MySQL puede utilizar phpMyadmin : <https://phpmyadmin.ovh.net/>

Para bases de datos PostgreSQL puede utilizar phpPgadmin : <https://phppgadmin.ovh.net/>

Para servidores SQL privados, puede utilizar phpMyadmin privado : <https://sqlprive.ovh.net/>

NOTA : Si ha olvidado la contraseña de su base de datos, puede cambiarla a través de su espacio Manager.

Si la conexión a través del gestor en línea funciona, podemos pasar a configurar un script con dichos parámetros.

Si la conexión a través del gestor en línea no funcionara y diera un error, revise sus parámetros previamente ya que no podrá continuar sin haber verificado que los parámetros son correctos.

Declaración de variables

La primera etapa consiste en declarar las variables que permitan la conexión con la base de datos (son los parámetros de las funciones de conexión a la base). Estas variables son:

```
$host : servidor SQL
```

```
$bdd : nombre de la base de datos
```

```
$user : nombre de usuario
```

```
$passwd : contraseña
```

Recuerde: Las variables no son visibles por los visitantes, dado que el script(.php) es sistemáticamente interpretado por el servidor Web

Estas variables son de hecho los parámetros de la función que permiten la conexión a la base. Así que es posible pasar los valores de estas variables directamente a cada función permitiendo la conexión, aunque esto puede ser un inconveniente después de un cambio de password por ejemplo. Lo ideal es guardar esos valores en un fichero aparte al que puedan acceder el resto de ficheros y así al querer modificar sólo se tenga que hacer una vez.

Variable \$host / servidor SQL

Normalmente el host tiene la forma sql5-XX. Dicho nombre de servidor SQL se le envía por correo durante la creación de la base de datos MySQL y PostgreSQL. Si tiene un servidor SQL Privado, en cambio debe poner la IP del servidor SQL.

Ejemplos :

Servidor SQL sql5-21

- \$host = 'sql5-21'

Servidor SQL privado con IP 10.0.216.31

- \$host = '10.0.216.31'

Variable \$bdd/ nombre de la base de datos

El nombre de la base de datos se establece al crear dicha base de datos. En el Manager está limitado a ciertos nombres si utiliza bases de datos MySQL o PostgreSQL, ya que todos los nombres de las bases de un alojamiento deben empezar con las mismas letras.

En las bases de datos de los SQL privados, sin embargo, tiene libertad completa para establecer el nombre de las bases que desee, siempre teniendo en cuenta que no puede crear dos bases de datos con el mismo nombre.

Variable \$user / nombre de usuario

El nombre de usuario o login, es el nombre de usuario que realiza la conexión. *Normalmente el login y el nombre de la base de datos es el mismo* En algunos casos, como en el SQL privado, usted puede decidir que sean diferentes.

Variable \$password / contraseña

Tanto en las bases de datos del Manager, MySQL y PostgreSQL, como en las bases de datos del SQL Privado, usted puede definir y cambiar a su gusto, la contraseña que desee para su base de datos.

Conexión a la base MySQL con gestión de errores

Ciertas de estas funciones reenvían un valor que permite conocer el estado de la conexión, de este modo es posible interrumpir el script para evitar los errores en cascada. Dos métodos permiten efectuar esta operación:

El almacenamiento del resultado de la ejecución de la función en una variable.

Por ejemplo:

```
$connect = mysql_connect($host,$user,$passwd);
```

El uso de la función die() en caso de error de ejecución. Si la función devuelve el valor 0 (es decir que hay un error) la función die() (se traduce muere) reenvía un mensaje de error.

Por ejemplo:

```
mysql_connect($host,$user,$passwd)  
or die("error de conexión al servidor $host");
```

Conexión a la base PostgreSQL con gestión de errores

Ciertas de estas funciones reenvían un valor que permite conocer el estado de la conexión, de este modo es posible interrumpir el script para evitar los errores en cascada. Dos métodos permiten efectuar esta operación:

El almacenamiento del resultado de la ejecución de la función en una variable.

Por ejemplo:

```
$connect = pg_connect("host=".$host." dbname=".$bbdd." user=".$user."  
password=".$password);
```

El uso de la función die() en caso de error de ejecución. Si la función devuelve el valor 0 (es decir que hay un error) la función die() (se traduce muere) reenvía un mensaje de error.

Por ejemplo:

```
pg_connect("host=".$host." dbname=".$bbdd." user=".$user."  
password=".$password);
```

```
or die('No se puede conectar: ' . pg_last_error());
```

Tratamiento de los resultados

Cuando efectuamos una consulta de selección de tuplas con la ayuda de la función `mysql_query`, es esencial almacenar el resultado de la consulta (los registros del resultado) en una variable, que solemos llamar `$result`.

Sin embargo, esta variable contiene el conjunto de los registros y no se puede utilizar tal cual. Hay que utilizar la función `mysql_fetch_row()`, que desglosa las líneas de resultado en columnas (por ejemplo Nombre, direccion...) y las mete en una variable de tabla en el orden en que se hayan recuperado.

Así, imaginemos una tabla llamada `enlace` que contenga el nombre y la URL de los sitios internet. Es posible recuperar el conjunto de los registros y añadirlos en una tabla:

```
<html>
<head>
<title>Enlaces
</head>

<body>
<table border="1" cellpadding="0" cellspacing="0">
<tr>
<th>Nombre del sitio</th>
<th>URL</th>
</tr>
<?php
Declaración de parámetros de conexión
$host = 'sql-XX';

$bbdd = 'su_base';
$user = 'su_login';

$passwd = password;
Conexión al servidor
mysql_connect($host, $user,$passwd) or die("error de conexión al
servidor");
mysql_select_db($bbdd) or die("error de conexión a la base de datos");

Creación y envío de la consulta
$query = "SELECT nombre,url FROM sitios ORDER BY nombre";
$result = mysql_query($query);
```

```

Recuperación de resultados
while($row = mysql_fetch_row($result)){
    $Nom = $row[0];
    $Url = $row[1];
    echo "<tr>
    <td><a href=\"$Url\">$Nom</td>
    <td>$Url</td>
    </tr>";
}
Desconexión de la base de datos
mysql_close();
?>
</tr>
</table>
</body>
</html>

```

En el ejemplo de arriba, las consultas devuelven los campos nombre y url. La función `mysql_fetch_row()` analiza cada línea de resultado de la consulta y almacena las columnas en la tabla `row[]`. Así, el campo nombre será almacenado en `row[0]` y url en `row[1]`. Por otro lado, incluimos normalmente `mysql_fetch_row()` en un bucle `while` para de esta forma poder tratar el conjunto de líneas de resultado. Cuando no hay más registros que recuperar(fetch), el bucle `while` finaliza y el intérprete ejecuta las siguientes instrucciones.

Detectar un resultado nulo

Puede ser útil, antes de insertar datos en una tabla, detectar la presencia de un registro en la tabla, para evitar duplicaciones de almacenamiento. Esto se puede hacer efectuando una consulta SQL con una orden `SELECT` y una cláusula `WHERE` que permite verificar la presencia o no de registros correspondientes a la consulta. La no presencia de resultado se traduce por un valor nulo devuelto por parte de la función `mysql_fetch_row()`. Aquí un ejemplo que añade el resultado de una consulta en caso favorable, y en caso contrario una frase que explique que no se ha encontrado ningún registro correspondiente(el código HTML en el cual el código PHP debe ser implantado ha sido omitido):

```

Declaración de parámetros de conexión
$host = la_maquina;

$bbdd = su_base;
$user = su_login;

$password = Password;

```

OVH

Conexión al servidor

```
mysql_connect($host, $user,$passwd) or die("error de conexión al
servidor");
mysql_select_db($bdd) or die("error de conexión a la base de datos");
```

Creación y envío de la consulta

```
$query = "SELECT nombre,url FROM sitios ORDER BY nombre";
$result = mysql_query($query);
```

Recuperación de resultados

```
if (!mysql_fetch_row($result)) {
echo "No se ha recuperado ningún registro";
}
else {
while($row = mysql_fetch_row($result)){
$Nom = $row[0];
$url = $row[1];
echo "<tr>
<td><a href=\"$url\">$Nom</a></td>
<td>$url</td>
</tr>";
}
}
```

Desconexión de la base de datos

```
mysql_close();
?>
```

Más información