

MySQL-PHP-Tutorial:

Datenbankverbindung mit PHP und MySQL:

Aufgabenstellung:

Es existiert eine MySQL-Datenbank mit einer Tabelle. Auf diese soll ein PHP-Skript zugreifen und Daten auslesen. In diesem Fall ist es eine kleine Adressdatenbank (Tabelle adressen) mit den Feldern id, anrede, vorname, nachname, strasse, plz, wohnort, fon, fax, email. Die Tabelle soll auf einer Webseite ausgegeben werden.

Lösung:

Schritt 1: HTML-Tabelle anlegen

```
<table cellpadding="1" cellspacing="3" border="1">
  <tr>
    <td>ID</td>
    <td>Anrede</td>
    <td>Vorname</td>
    <td>Nachname</td>
    <td>Strasse</td>
    <td>PLZ</td>
    <td>Wohnort</td>
    <td>Telefon</td>
    <td>Fax</td>
    <td>Email</td>
  </tr>
  <tr>
    <td></td>
    <td></td>
    <td></td>
    <td></td>
    <td></td>
    <td></td>
    <td></td>
    <td></td>
    <td></td>
    <td></td>
  </tr>
</table>
```

Schritt 2: Verbindung mit dem MySQL-Server herstellen

Jetzt haben wir eine statische Tabelle erstellt, die wir manuell Zeile um Zeile füttern könnten. Aber PHP ist dynamisch und nimmt über den Befehl `mysql_connect()` eine Verbindung mit dem MySQL-Server auf und wählt über `mysql_select_db()` eine Datenbank aus. Der Funktion `mysql_connect()` übergibt man mehrere Parameter:

- 1.) den mysql-Host speichern wir in der Variablen `$mysqlhost`
- 2.) den mysql-User speichern wir in der Variablen `$mysqluser`
- 3.) das mysql-Passwort speichern wir in der Variablen `$mysqlpwd`

```
$mysqlhost="localhost"; // MySQL-Host angeben
$mysqluser="user1"; // MySQL-User angeben
$mysqlpwd="abcd1234"; // Passwort angeben
```

Die Variablen werden durch Komma getrennt der Funktion als Parameter übergeben und die Kennung in einer Variablen `$connection` gespeichert:

```
$connection=mysql_connect($mysqlhost, $mysqluser, $mysqlpwd) or die
("Verbindungsversuch fehlgeschlagen");
```

Falls die der MySQL-Server ausfällt oder die eingegeben Daten falsch waren wird der Teil hinter "or" ausgeführt: die Funktion `die()` erwartet einen String, und gibt diesen als Textnachricht aus, falls die Verbindung zum Server nicht

erfolgreich war. Das komplette Skript würde dadurch abgebrochen werden und die Datei wird nicht weiterverarbeitet (geparst).

Schritt 3: Verbindung mit der Datenbank herstellen

Mit der Funktion `mysql_select_db()` wird eine Verbindung zur gewünschten Datenbank hergestellt. Der Name dieser wird am besten auch in einer Variablen `$mysqldb` abgelegt, so dass alle Verbindungsparameter einfach geändert werden können (z.B. beim Umzug auf einen anderen Server). Die MySQL-Server-Kennung `$connection` wird als zweiter Parameter der Funktion `mysql_select_db()` übergeben.

```
$mysqldb="db_user1"; // Gewuenschte Datenbank angeben
mysql_select_db($mysqldb, $connection) or die("Konnte die Datenbank nicht
waehlen.");
```

Schritt 4: Auswahl der Zeilen aus der Tabelle

Im nächsten Schritt wird SQL gesprochen, die Sprache mit der wir mit der Datenbank kommunizieren. Die Anweisung stellt einen einfachen Text-String dar, der in einer Variablen `$sql` gespeichert wird.

```
$sql = "SELECT id, anrede, vorname, nachname, strasse, plz, wohnort,
fon, fax, email FROM adressen";
```

SELECT wird benutzt, um ausgewählte Zeilen aus einer Tabelle abzurufen. Statt 'id, anrede, vorname, nachname, strasse, plz, wohnort, fon, fax, email' kann man auch '*' schreiben, wenn alle Felder ausgewählt werden sollen. Weitere Infos zur SELECT-Syntax unter der vorzüglichen MySQL-Referenz unter <http://dev.mysql.com/doc/mysql/de/SELECT.html>.

Schritt 5: Anfrage an MySQL senden

Die Funktion `mysql_query()` sendet eine Anfrage an mysql, wobei dieser der String `$sql` als Parameter übergeben wird. Bei einer SELECT-Anweisung wird bei Erfolg eine Ressourcen-Kennung ausgegeben, die in einer Variablen `$adressen_query` gespeichert wird.

```
$adressen_query = mysql_query($sql) or die("Anfrage nicht erfolgreich");
```

Schritt 6: Anzahl der Datensätze ausgeben

Mit dieser Ressourcen-Kennung lassen sich jetzt schöne Sachen anstellen. Z.B. kann bei unserer SELECT-Anweisung mit `mysql_num_rows()` die Anzahl der betroffenen Zeilen ausgegeben werden:

```
$anzahl = mysql_num_rows($adressen_query);
echo "Anzahl der Datensätze: $anzahl";
```

Schritt 7: Assoziatives Array erzeugen

Oder man kann mit `mysql_fetch_array()` ein Array mit den betroffenen Datensätzen erzeugen. Dies machen wir uns zu nutzen, in dem wir die Daten des Arrays über eine while-Schleife auslesen:

```
while ($adr = mysql_fetch_array($adressen_query)) {
}
```

Da es sich um ein assoziatives Array handelt, kann man innerhalb der while-Schleife auf die Werte der Array-Elemente zugreifen und diese ausgeben.

Schritt 8: Tabellenzeilen dynamisch generieren

In der while-Schleife lässt sich auch das Tabellen-Konstrukt in HTML generieren. Wir ersetzen einfach die zweite Zeile der statischen Tabelle durch die PHP-Anweisung:

```
<table cellpadding="1" cellspacing="3" border="1">
  <tr>
```

```

        <td>ID</td>
        <td>Anrede</td>
        <td>Vorname</td>
        <td>Nachname</td>
        <td>Strasse</td>
        <td>PLZ</td>
        <td>Wohnort</td>
        <td>Telefon</td>
        <td>Fax</td>
        <td>Email</td>
    </tr>
<?php
while ($adr = mysql_fetch_array($adressen_query)) {
?>
    <tr>
        <td></td>
        <td></td>
        <td></td>
        <td></td>
        <td></td>
        <td></td>
        <td></td>
        <td></td>
        <td></td>
        <td></td>
    </tr>
<?php
}
?>
</table>

```

In die einzelnen Tabellen-Zellen setzen wir die Werte des assoziativen Arrays. Hier nur der dynamisch betroffene HTML-Tabellenteil zwischen <tr> und </tr>:

```

<?php
while ($adr = mysql_fetch_array($adressen_query)) {
?>
    <tr>
        <td><?=$adr['id']?></td>
        <td><?=$adr['anrede']?></td>
        <td><?=$adr['vorname']?></td>
        <td><?=$adr['nachname']?></td>
        <td><?=$adr['strasse']?></td>
        <td><?=$adr['plz']?></td>
        <td><?=$adr['wohnort']?></td>
        <td><?=$adr['fon']?></td>
        <td><?=$adr['fax']?></td>
        <td><?=$adr['email']?></td>
    </tr>
<?php
}
?>

```

"<?=" ist eine Kurzschreibweise für "<?php echo" und lässt den Quelltext von eingestreuten PHP-Fragmenten in HTML übersichtlicher wirken.

Fertig ist unsere Tabelle. Wer will kann sie mit HTML und CSS noch verschönern.

Das komplette Listing:

```

<?php
$mysqlhost="localhost"; // MySQL-Host angeben
$mysqluser="user1"; // MySQL-User angeben
$mysqlpwd="abcd1234"; // Passwort angeben
$mysqlpdb="db_user1"; // Gewuenschte Datenbank angeben

$connection=mysql_connect($mysqlhost, $mysqluser, $mysqlpwd) or die
("Verbindungsversuch fehlgeschlagen");

```

```

mysql_select_db($mysqldb, $connection) or die("Konnte die Datenbank nicht
waehlen.");

$sql = "SELECT id, anrede, vorname, nachname, strasse, plz, wohnort, fon, fax,
email FROM adressen";

$adressen_query = mysql_query($sql) or die("Anfrage nicht erfolgreich");

$anzahl = mysql_num_rows($adressen_query);
echo "Anzahl der Datensätze: $anzahl";
?>

<table cellpadding="1" cellspacing="3" border="1">
  <tr>
    <td>ID</td>
    <td>Anrede</td>
    <td>Vorname</td>
    <td>Nachname</td>
    <td>Strasse</td>
    <td>PLZ</td>
    <td>Wohnort</td>
    <td>Telefon</td>
    <td>Fax</td>
    <td>Email</td>
  </tr>

<?php
while ($adr = mysql_fetch_array($adressen_query)) {
?>
  <tr>
    <td><?=$adr['id']?></td>
    <td><?=$adr['anrede']?></td>
    <td><?=$adr['vorname']?></td>
    <td><?=$adr['nachname']?></td>
    <td><?=$adr['strasse']?></td>
    <td><?=$adr['plz']?></td>
    <td><?=$adr['wohnort']?></td>
    <td><?=$adr['fon']?></td>
    <td><?=$adr['fax']?></td>
    <td><?=$adr['email']?></td>
  </tr>
<?php
}
?>

</table>

```

Links zum Thema:

PHP.net MySQL-Referenz in deutsch

<http://de.php.net/manual/de/ref.mysql.php>

//© 2004 by Gurkcity Webdesign - www.gurkcity.de