Erstellen von dynamischen Webseiten mit

PHP 3 *my*SQL

Andreas Hahn

Projektgruppe Praxisorientierte Informatik

Hochschule für Technik und Wirtschaft des Saarlandes

Goebenstr. 40, 66117 Saarbrücken

Inhalt

1. Erstellen von dynamischen Webseiten mit PHP3 und MySQL

1.1 Datenbankprogrammierung unter MySQL

MySQL ist eine relationale Datenbank, die auf Linux-, Solaris- oder Windows-Plattformen läuft. Die am häufigsten verwendeten Plattformen sind Linux und Solaris. Im Zusammenhang mit der Programmiersprache PHP/FI kann über das Internet auf Datenbanken zugegriffen werden.

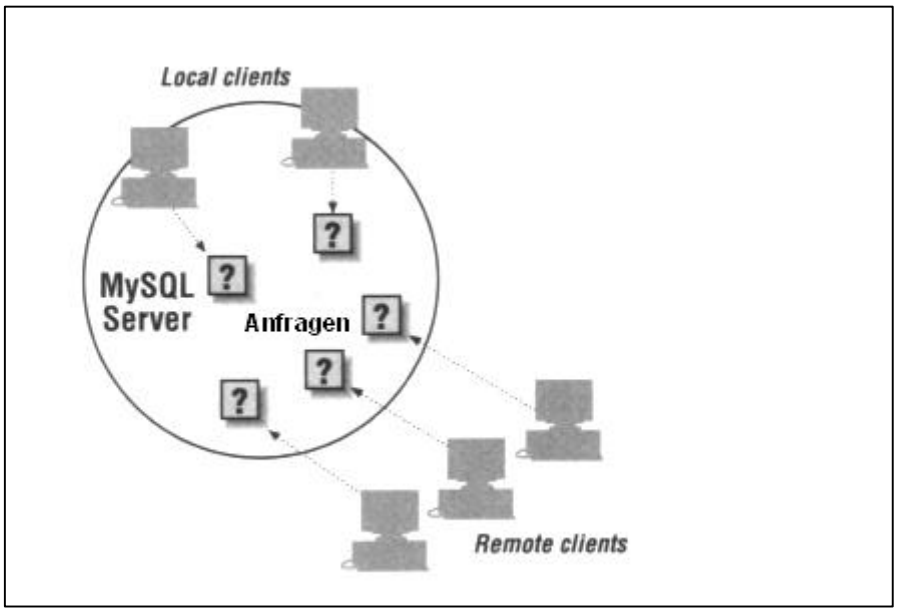


Abbildung 1: Client-/Server- Architektur einer MySQL Datenbank

[Quelle: MySQL&msql S. 40]

1.1.1 Einrichten von MySQL unter Linux

Aktuelle Versionen von MySQL können von folgender Internetseite kostenlos bezogen werden: http://www.mysql.com

Verschiedene Linuxversionen beinhalten bereits eine MySQL-Distribution, so z.B. S.u.S.e-Linux.

Nach der Installation von MySQL sollte das Passwort des Administrators geändert werden. Der Befehl dazu lautet:

```
mysqladmin –u root password  ’meinneuespasswort’
```

Die Änderung des Passwortes hat den Vorteil, dass niemand ausser dem Administrator, Änderungen an bestehenden Datenbankdefinitionen vornehmen kann. Es ist ohne Passworteingabe nicht möglich, eine neue Datenbank aufzubauen.

1.1.1.1 Datenbankadministration

Die Verwaltung von MySQL lässt sich mit dem Tool **mysqladmin** komfortabel gestalten. Mysqladmin ist ein mächtiges Werkzeug zum Erstellen oder Löschen von Datenbanken.

Erstellen, bzw. Löschen einer Datenbank:

Neue Datenbanken werden mit dem Befehl:

```
mysqladmin –p create DATENBANKNAME
```

erstellt. –p steht für das Passwort des Administrators steht. Wenn das richtige Passwort eingegeben wurde, wird eine neue, leere Datenbank angelegt.

Da eine Datenbank unter MySQL eine Menge von Dateien in einem bestimmten Verzeichnis darstellt, generiert der Befehl mysqladmin ein neues Verzeichnis, um die Dateien der neuen Datenbank zu speichern. Wenn z.B. eine Datenbank namens „boersen“ erstellt wird, legt MySQL in seinem Hauptverzeichnis ein Unterverzeichnis mit dem Namen boersen an.

Die Löschfunktion ist analog zur Erstellung einer Datenbank, mit dem Befehl:

```
mysqladmin –p drop DATENBANKNAME
```

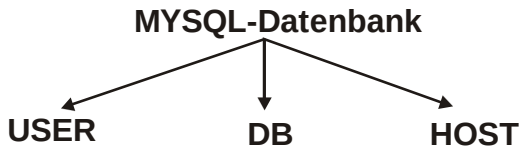
Mit diesem Befehl wird die Datenbank vollständig gelöscht.

1.1.1.2 Zugriffsrechte auf eine MySQL-Datenbank erteilen

Um auf eine Datenbank zuzugreifen, muss sie für bestimmte Benutzer zugänglich gemacht werden.

Die Verwaltung der Zugriffe auf Datenbanken unter MySQL wird durch die Datenbank **mysql** realisiert. In dieser Datenbank werden Zugriffsinformationen aller Datenbanken eingetragen. Es wird festgelegt, welche Anwender auf welche Datenbank zugreifen dürfen.

Aufbau der Datenbank mysql:



Die Datenbank mysql besteht folgenden drei Tabellen:

- **USER →** In der Tabelle **user** werden die Rechte der auf die Datenbanken zugreifenden Benutzer vergeben. Die erteilten Benutzerrechte in der Tabelle **user** gelten für alle Datenbanken, sobald in der Tabelle **db** keine gesonderte Zuordnung zwischen Benutzer und Datenbank erstellt wurde.

- **DB →** In der Tabelle **db** werden die Namen der erstellten Datenbanken festgelegt.
  Desweiteren ist es möglich, Benutzerrechte auf Datenbankebene zu erteilen.

- **HOST →** In der Tabelle **host** wird festgelegt, welcher Server auf welche Datenbank zugreifen darf.

Im Folgenden werden die Tabellen der Datenbank **mysql** näher beschrieben.

Tabelle DB:

Feld	**Typ**	**Schlüssel**	**Default**
Host	Char(60)	Primärschlüssel	
Db	Char(32)	Primärschlüssel	
User	Char(16)	Primärschlüssel	
Select_priv	Enum('N','Y')		N
Insert_riv	Enum('N','Y')		N
Update_priv	Enum('N','Y')		N
Delete_priv	Enum('N','Y')		N
Create_priv	Enum('N','Y')		N
Drop_priv	Enum('N','Y')		N
References_priv	Enum('N','Y')		N
Index_priv	Enum('N','Y')		N
Alter_priv	Enum('N','Y')		N

Tabelle HOST:

Feld	**Typ/Größe**	**Schlüssel**	**Default**
Host	Char(60)	Primärschlüssel	
Db	Char(32)	Primärschlüssel	
Select_priv	Enum('N','Y')		N
Insert_riv	Enum('N','Y')		N
Update_priv	Enum('N','Y')		N
Delete_priv	Enum('N','Y')		N
Create_priv	Enum('N','Y')		N
Drop_priv	Enum('N','Y')		N
References_priv	Enum('N','Y')		N
Index_priv	Enum('N','Y')		N
Alter_priv	Enum('N','Y')		N

Tabelle USER:

Feld	**Typ/Größe**	**Schlüssel**	**Default**
Host	Char(60)	Primärschlüssel	
User	Char(16)	Primärschlüssel	
Password	Char(16)		
Select_priv	Enum('N','Y')		N
Insert_riv	Enum('N','Y')		N
Update_priv	Enum('N','Y')		N
Delete_priv	Enum('N','Y')		N
Create_priv	Enum('N','Y')		N
Drop_priv	Enum('N','Y')		N
Reload_priv	Enum('N','Y')		N
Shutdown_priv	Enum('N','Y')		N
Process_priv	Enum('N','Y')		N
File_priv	Enum('N','Y')		N
Grant_priv	Enum('N','Y')		N
References_priv	Enum('N','Y')		N
Index_priv	Enum('N','Y')		N
Alter_priv	Enum('N','Y')		N

Erklärung zu den einzelnen Feldern:

Host	→	Hostname, z.B. localhost
User	→	Benutzername
Password	→	Benutzerpasswort
Select_priv	→	Nach bestimmten Datensätzen suchen
Insert_priv	→	Neue Datensätze hinzufügen
Update_priv	→	Ändern von Datensätzen
Delete_priv	→	Datensätze löschen
Create_priv	→	Anlegen neuer Tabellen oder Datenbanken
Drop_priv	→	Löschen von Tabellen / Datenbanken
Reload_priv	→	Neustart der Datenbank
Shutdown_priv	→	Datenbank herunterfahren
Process_priv	→	Serverprozesse managen
File_priv	→	Lesen oder Schreiben von Dateien mit folgenden Befehlen SELECT INTO OUTFILE und LOAD DATA INFILE
Grant_priv	→	Zugriff auf Tabellen anderer Benutzer
Index_priv	→	Erstellen von Indizes
Alter_priv	→	Absetzen des Befehls ALTER TABLE

Um auf Datenbanken im MySQL-System zuzugreifen, müssen zunächst Zuordnungen zwischen Datenbankname, Hostname und Benutzereigenschaften hergestellt werden. Die Zuordnung erfolgt in der Datenbank **mysql**.

Beispiel:

Das folgende Beispiel zeigt eine Zuordnung zwischen Datenbankname, Hostname und Benutzereigenschaften.

Datenbankname	= boersen	→	der Datenbankname **boersen** wird in die Tabellen **db** und **host** eingetragen.
Hostname	= localhost	→	der Hostname **localhost** wird in die Tabellen **host** und **user** eingetragen.

Benutzername	= user	→	der Benutzername user wird in die Tabellen **user** und **db** eingetragen.
Benutzerpasswort	= user	→	das Benutzerpasswort **user** wird in die Tabelle **user** eingetragen.

In der Tabelle **db** wird der Datenbank **boersen** der Host **localhost** und Benutzer **user** zugeordnet. Desweiteren werden Zugriffsberechtigungen erteilt. In diesem Beispiel werden die Berechtidungen auf 'Y' gesetzt. D.h. der Benutzer **user** erhält alle Rechte an der Datenbank **boersen**.

```
insert into db values
('localhost','boersen','user','Y','Y','Y','Y','Y','Y','Y','Y','Y',
'Y');
```

In der Tabelle **host** wird dem Hostnamen **localhost** die Datenbank **boersen** zugeordnet. Der Host **localhos**t erhält vollen Zugriff auf die Datenbank **boersen**.

```
insert into host values
('localhost','boersen','Y','Y','Y','Y','Y','Y','Y','Y','Y','Y');
```

In der Tabelle **user** wird dem Hostnamen ein Benutzer zugeordnet. Der Benuzter **user** erhält vollen Zugriff auf den Host **localhost**.

Passwörter werden mit dem Syntax **password('passwort')** in die Tabelle **user** eingetragen.

```
insert into user values
('localhost','user',password('user'),'Y','Y','Y','Y','Y','Y','Y','Y'
,'Y','Y','Y','Y','Y','Y');
```

1.1.1.3 Tabellen einrichten

Bevor die Datenbank mit Werten gefüllt wird, sollte die Grundstruktur des Datenbankkonzeptes klar sein.

Im Allgemeinen wird zu Beginn ein sogenanntes „Entity-Relationship Modell (ER)“ erstellt, das Aufschlüsse über das Verhalten der Beziehungen zwischen den Objekten gibt. Desweiteren sind theoretische Vorüberlegungen bezüglich der Datenstruktur der Tabellen zu treffen.

Unter MySQL können Tabellen mit folgender Vorgehensweise erstellt werden:

1) Starten von mysql mit dem Namen der entsprechenden Datenbank.
   `mysql datenbankname`

   Beispiel: `mysql boersen`

2) Erstellen einer Tabelle
   `CREATE TABLE tabellen_name (create_definition,...)`

   Beispiel zum Erstellen einer Tabelle:

```
CREATE TABLE gewerk
(
       gewerk_nr      int(4)          DEFAULT '0' NOT NULL
     , gewbeschr      char(100)       DEFAULT '0' NOT NULL
     , merker         char(1)         DEFAULT 'N' NOT NULL
     , PRIMARY KEY    (gewerk_nr)
     , KEY gewbeschr (gewbeschr)
);
```

Jede Tabelle wird durch drei Dateien im Datenbankverzeichnis dargestellt:

tabellen_name.frm - Tabellendefinition
tabellen _name.isd - Datenfile
tabellen _name.ism - Indexfile

Bedeutung der drei Dateien:

- In der *Tabellendefinition* befindet sich die Struktur der Datenbank, z.B. die Angaben über die Größen der Felder.

- Im *Datenfile* befinden sich alle Daten, die dort so abgelegt sind, dass die schnell wiedergefunden werden können.

- Im *Indexfile* werden die Schlüsselbegriffe so abgelegt, dass eine schnelle Suche im Datenfile ermöglicht wird. Zur Anwendung kommt ein Verfahren, das es ermöglicht, die genauen Speicherorte von Datenbankeinträgen zu ermitteln. Bei Personennamen beispielsweise, werden die Buchstaben des Namens errechnet. Dies geschieht, indem die ASCII-Werte der Buchstaben Modulo einer Primzahl, die größer als die Anzahl der Datenbankeinträge ist, addiert wird.

Erstellung von Tabellen mittels DUMP-Datei:

Zur Erleichterung der Datenbankerstellung bietet MySQL eine sogenannte Dump-Möglichkeit. Mit dem Befehl mysqldump kann die Struktur und der Inhalt einer Datenbank in eine Datei kopiert werden.

```
mysqldump datenbankname > dumpdatei.dump
```

Der umgekehrte Weg ist auch möglich. Die Datenbankstruktur wird in einer Datei abgespeichert und mit folgendem Befehl in die Datenbank übernommen:

```
mysql datenbankname < dumpdatei.dump
```

Da mit dem Befehl mysqldump die komplette Datenbankstruktur und der Inhalt der Tabellen in eine Datei kopiert wird, eignet sich dieses Verfahren ideal zum Backup von Datenbanken.

Beispiel:

```
mysql boersen < /usr/local/mysql/bin/boersen.dump
```

Zum Hochladen der Dump-Datei wird der komplette Pfad der Datei angegeben.

1.2 Aufbau von PHP/FI

Zur Internetprogrammierung wird die Skriptsprache PHP/FI (Personal/Professional Homepage Construction Kit / Form Interpreter) eingesetzt. PHP/FI eignet sich zur Anbindung an Internetdatenbanken, wie beispielsweise MySQL.

1.2.1 Softwareunterstützung durch PHP/FI

PHP/FI besitzt die Möglichkeit auf verschiedene Datenbanken zuzugreifen.

Zur Zeit werden von PHP folgende Datenbanken unterstützt:

- Adabas D
- InterBase
- Solid
- dBase
- mSQL
- Sybase
- Empress
- MySQL
- Velocis
- FilePro
- Oracle
- Unix dbm
- Informix
- PostgreSQL.

Folgende Protokolle werden von PHP unterstützt :

- IMAP
- SNMP
- NNTP
- POP3
- HTTP.

1.2.2 Aufbau und Funktionsweise von PHP/FI

In der Web-Entwicklung unterscheidet man grundsätzlich zwischen c*lientseitiger* und s*erverseitiger* Programmierung. PHP-Anwendungen laufen auf dem Server ab.
Eine typische Kommunikation zwischen Browser und Web-Server sieht so aus (s. auch Abbildung 2):

1. Der Browser fordert beispielsweise eine PHP3-Seite vom Webserver an.

2. Der Webserver übergibt die Kontrolle an den PHP-Interpreter.

3. Der Webserver bzw. das zuständige Modul (z. B. mod_PHP) liefert die angeforderten Daten zurück und schickt sie als HTML-Datei zum Browser.

Da PHP eine Skriptsprache ist, wird ein Interpreter ("Übersetzer") benötigt, der den Quellcode analysiert und ausführt. Der Interpreter kann direkt in den Web-Server eingebunden (als Apache- oder mit PHP 4.0 als NSAPI/ISAPI-Modul) oder als eigenes Programm für eine CGI-Umgebung eingesetzt werden. Bei jedem Aufruf einer Webseite durchläuft der Interpreter das PHP-Skript und reicht Ausgaben an den Browser weiter. Die Integration in den Webserver bringt einen deutlichen Geschwindigkeitsvorteil, da nicht bei jeder Seitenanfrage ein externes Programm gestartet werden muß, wie es zum Beispiel bei herkömmlichen Perl-CGI-Skripten der Fall ist.

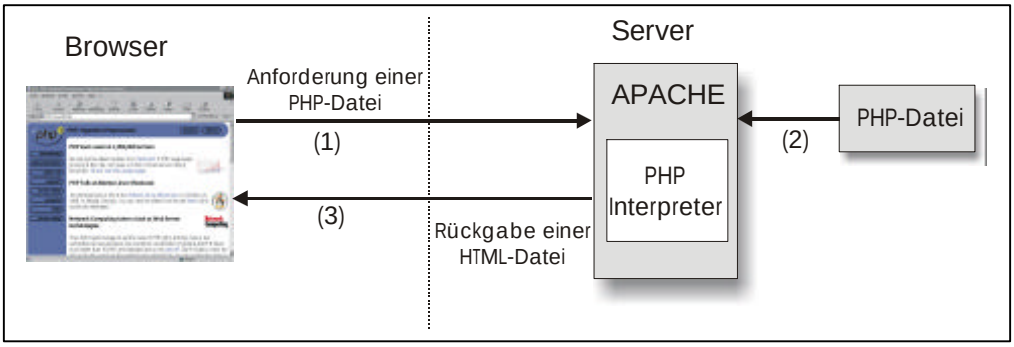


Abbildung 2: Aufbau einer Verbindung zwischen Browser und Server

1.2.3 PHP in HTML einbinden

„PHP wird im Gegensatz zu den normalen CGI-Programmen nicht als eigenständiges Programm ausgeführt. Der fundamentale Unterschied liegt darin, dass nicht HTML in die Programmiersprache eingebettet wird, sondern PHP in HTML eingebunden wird. Typischerweise heißen die Dateien dann nicht mehr `.html` oder `.htm`, sondern `.php3`. An dieser Dateiendung erkennt der Webserver, dass es sich um eine in PHP geschriebene Datei handelt.“ [Schmid/Cartus/Blume, S. 40ff]

Um den Unterschied zu XML-konformen (eXtensible Markup Language) Tags darzustellen, beginnt der PHP-Code mit **`<?PHP`** und endet mit **`?>`**.
Zu beachten ist, dass jede PHP-Zeile mit einem Semikolon (`;`) abgeschlossen wird.

Es gibt zwei Möglichkeiten, PHP-Code in HTML einzubinden:

1) Die Einbindung von PHP3 in HTML-Dokumente erfolgt mit folgender Syntax:
   `<?PHP ... ?>`

   In dieser Darstellung wird das HTML-Dokument mit der Endung php3 angelegt.

Beispiel zur Einbindung von PHP in HTML:

```
<html>
      Normaler HTML-Code
<br>

<?php
      printf(“Ausgabe eines Textes mit PHP“);
?>

<br>
      HTML-Code
</html>
```

2) Damit auch ältere Browser, wie z.B. Netscape-Navigator 2.0 oder MS-Internet Explorer Version 3.0, PHP darstellen können, findet die sogenannte skriptkonforme Schreibweise Anwendung.

Beispiel zum Einbinden von PHP als Skript in HTML:

```
<html>
      Normaler HTML-Code
<br>

<script language="PHP">

      printf("Ausgabe eines Textes mit PHP") ;

</script>

<br>
      HTML-Code
</html>
```

Mit den Tags (Syntax) `<script language="PHP"> ... </script>` erhält der Browser die Information, dass in diesem Bereich PHP ausgeführt wird.

1.2.4 Variablen in PHP

Folgende Variablentypen werden von PHP unterstützt :

`Integer`	ganzzahlige Werte
`Double`	Fließkommawerte
`String`	ASCII-Zeichen zwischen 0 und 32768
`Array`	Ein- und mehrdimensionale, indizierte assoziative Arrays
`Objekt`	Instanz einer Klasse mit Eigenschaften und Methoden

Variablen können überall und zu jeder Zeit inintialisiert werden. Sie müssen nicht zu Beginn des Skripts deklariert werden.

Um den Überblick zu wahren, ist es allerdings sehr sinnvoll, alle Variablen zu Beginn des PHP-Quellcodes zu deklarieren.

Alle Variablennamen beginnen mit dem Dollar-Zeichen (**$**)
PHP unterscheidet Groß-Kleinschreibung!
D.h. **$x** ist nicht das gleiche wie **$X**.

Zur Initialisierung von Variablen, wird ihr einfach ein Wert zugewiesen. PHP wählt den richtigen Variablentyp.

Beispiel für eine Variablendeklaration:

```
<?php
      $a = 9;                  // PHP erkennt automatisch einen Integer-Wert
      $b = "Hallo Leute";      // PHP erkennt automatisch einen String
      $c = 10.5;               // PHP erkennt automatisch eine Fließkommazahl

      printf(„Ausgabe der Zahl: ", $a);
      printf(„Ausgabe des String: ", $b);
?>
```

1.2.5 Parameterübergabe mit PHP

1.2.5.1. Methode GET

Die Grundstruktur mit der GET-Methode in HTML sieht folgendermassen aus:

```
<form action="zieldatei" method="get">
      . . .
      weitere Anweisungen
      . . .
</form>
```

Die **GET**-Methode `method=get` erzwingt in einem HTML-Fomular, dass die ausgefüllten Formulardaten zuerst an die Server-Software übertragen und von dieser in einer bestimmten Umgebungsvariablen (QUERY_STRING) zwischengespeichert werden. Die im `action=`-Parameter angegebene Datei liest den Inhalt der Umgebungsvariablen aus, um dann an die Formulardaten heranzukommen. Der Formulardatenstrom hängt, getrennt durch ein Fragezeichen, direkt hinter der URL-Adresse der im `action=`-Parameter angegebenen Datei.
[Quelle: php-Buch S. 337]

Wenn die Zieldatei auf der URL: http://swlablinux.htw-saarland.de/warenhaus/zieldatei steht, sieht die Übertragung mit der GET-Methode folgendermassen aus:
http://swlablinux.htw-saarland.de/zieldatei?auswahl

Beispiel zur Methode GET:

Im folgenden Beispiel wird die Parameterübergabe mit der Methode GET näher dargestellt. Das Beispiel dient zur Addition von zwei Werten. Mit dem Übertragungsbutton werden die Werte an die Empfangdatei gesendet. In diesem Beispiel sind Sende- und Empfangsdatei das gleiche PHP3-Dokument. Wenn beispielsweise für den Wert 1 die Zahl 3 und für Wert 2 die Zahl 2 eingetragen wird, dann sieht die Übertragung der Parameter folgendermassen aus:

http://swlablinux.htw-saarland.de/warenhaus/beispiele/bsp_berechnen.php3?wert1=3&wert2=2

```
<head>
<title>Beispiel für Parameterübergabe mit Methode POST</title>
</head>
<body>

<?php
     // Variablendeklaration:
     $wert1;
     $wert2;
     $ergebnis;

     // Übertragungsmethode GET:
     printf("<form action=\"bsp_berechnen.php3\" method=\"GET\">");
     printf("<p>Bitte geben Sie zwei Werte ein:</p>");

     // Werteingabe:
     printf("<input name=\"wert1\"><br>");
     printf("<input name=\"wert2\"><br>");

     // Übertragungsbutton:
     printf("<input type=submit value=\"Berechnen!\">");
     printf("</form>");

     // Eingabeüberprüfung:
     if(!(empty($wert1)||empty($wert2)))
     {
          // Ergebnis berechnen:
          $ergebnis=$wert1+$wert2;

          // Ergebnis ausgeben:
          printf("<p>$wert1+$wert2=$ergebnis</p>");
     }
?>
</body>
</html>
```

1.2.5.2. Methode POST

Die Grundstruktur der POST-Methode in HTML sieht folgendermassen aus:

```
<form action="zieldatei" method="post">
      . . .
      weitere Anweisungen
      . . .
</form>
```

Die **POST**-Methode `method=post` überträgt die im HTML-Formular ausgefüllten Formulardaten direkt an die Adresse im `action=`-Parameter.
Um die Formularangaben auszulesen, wird die Standardeingabe ausgelesen. Der übergebene Datenstrom enthält, im Gegensatz zur GET-Methode, kein Datenendkennzeichen.
[Quelle: Jörg Krause S. 340]

Die POST-Methode wird analog zur GET-Methode dargestellt. Anstatt GET, wird im HTML-Formular POST eingesetzt: <form action="..." nethod="post">

1.3 Internetverbindung zu MySQL mittels PHP/FI

1.3.1. Verbindungsaufbau zu einer MySQL-Datenbank

PHP baut eine Verbindung zum Datenbankserver (Abbildung 3) auf und öffnet einen Kanal, um der Datenbank zu signalisieren, dass nun Anfragen folgen. Die Datenbank muß sich dabei nicht auf dem gleichen Webserver befinden. Alle wichtigen Datenbanken können über TCP/IP angesprochen werden.

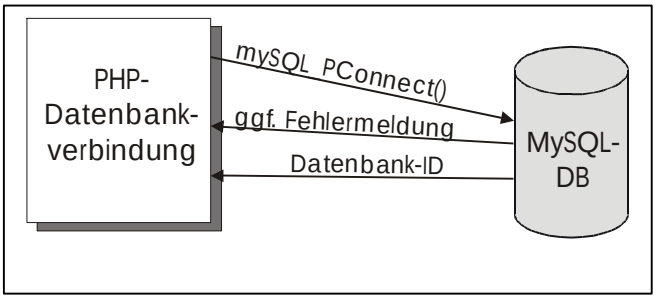


Abbildung 3: Verbindungsaufbau zu einder MySQL-Datenbank

Um eine mit PHP/FI eine Verbindung zu einer MySQL-Datenbank herzustellen, wird folgende Syntax verwendet:

```
mySQL_PConnect(hostname, benutzername, benutzerpasswort)
```

Obige Anweisung öffnet eine Verbindung zum Host hostname. Zur Anmeldung muss der Anwender seinen Benutzernamen sowie sein Passwort eingeben. Im Allgemeinen sind Benutzername und Passwort in der Datenbank mysql im MySQL-Datenbanksystem hinterlegt. Somit ist eine Authentifizierung nicht erforderlich.

Beispiel für einen Verbindungsaufbau:

```
<?php
      mysql_pconnect(“localhost”, “user”, “user”);
?>
```

1.3.2. Anweisungen an die MySQL-Datenbank senden

1.3.2.1. Select-Operation

Eine Select-Operation mit PHP/FI kann wie folgt ausgeführt werden:

1) Verbindung zur Datenbank herstellen
2) Select-Operation absetzen
3) Ergebnismenge ermitteln
4) Ergebnismenge überprüfen
5) Daten oder entsprechende Fehlermeldung ausgeben.

Folgendes Beispiel soll Betriebsname und Anschrift zu einem Handwerksunternehmen aus Schmelz ausgeben.

Bedingungen:

a) Ausgabe ist nach Unternehmensnamen sortiert.

Vorgehensweise:

1) Verbindung zur Datenbank herstellen

```
mySql_PConnect("", "user", "user");
```

2) Select-Operation ausführen

```
$query = " SELECT betr_name, ort, plz, str_haus ";
$query = $query . " FROM stammdaten WHERE ort = 'Schmelz' ";
$query = $query . " AND sperre = 'N' ORDER BY betr_name";
```

3) Ergebnismenge initialisieren

```
$result = mysql("boersen", $query);
```

4) Ergebnismenge überprüfen

Wenn die Ergebnismenge > 0, schreibe die Anzahl der ermittelten Felder in die Variable $Gesamt. mysql_Num_Rows($result) liefert die Anzahl der Datensätze in der Ergebnismenge.

```
if (mysql_Num_Rows($result) > 0)
{
     $Gesamt = mysql_NumRows($result);
}
else
{
     $Gesamt = 0;
}
```

Ist die Ergebnismenge leer, so wird eine entsprechende Meldung ausgegeben.

```
if ($Gesamt == 0)
{
     printf("Es stehen keine Daten zur Verfügung");
}
```

Ansonsten Ausgabe der ermittelten Werte.

```
else
{
     printf("<table border=0>");
     printf("<tr>");
     printf("<th>Betriebsname</th>");
     printf("<th>Anschrift</th>");
     printf("</tr>");
}
```

Mit der While-Schleife werden die Datensätze aus der Datenbank gelesen. Mit dem Befehl printf(); werden die einzelnen Datensätze auf dem Bildschirm ausgegeben.

```
$i = 0;
while ($i < mysql_NumRows($result))
{
     $betr_name= mysql_Result($result, $i, "betr_name");
     $str_haus = mysql_Result($result, $i, "str_haus");
     $plz = mysql_Result($result, $i, "plz");
     $ort = mysql_Result($result, $i, "ort");
     printf("<tr>");
     printf("<th>%s</th>", $betr_name);
     printf("<th>");
     printf($str_haus);
     printf("<br>");
     printf($plz . $ort);
     printf("</th>");
     printf("</tr>");
$i++;
}
printf("</table>");
```

1.3.3. Fehlerbehandlung bei gescheitertem Verbindungsaufbau

MySQL bietet standardmässig Fehlermeldungen an. Eine detaillierte Auflistung der Fehlernummern und deren Verhalten befinden sich in den Bibliotheken `mysqld_e.h`, `mysys_err.h`.

Eine Überprüfung des Verbindungsaufbaus ist möglich mit mysql_errno und mysql_error.

1.3.3.1. mysql_errno / mysql_error

Im folgenden Beispiel wird versucht, eine Verbindung zu einem Host aufzubauen. Wenn weder Host, Datenbank oder Tabelle gefunden werden, erscheint eine Fehlernummer mit entsprechendem Fehlertext.

```
<?php
      mysql_pconnect($host);
      echo mysql_errno() . ": ". mysql_error() ."<br>";
      mysql_select_db("nichtexistierendeDB");
      echo mysql_errno() . " : ". mysql_error() . "<br>";
      $query=mysql_query("SELECT * FROM nichtexistierendeDB");
      echo mysql_errno() . " : ". Mysql_error() . "<br>";
?>
```

1.3.4. Zusammenfassung

Die wichtigsten PHP-Befehle zum Verbindungsaufbau zu einer MySQL-Datenbank:

<table>
<tr>
<td>mysql_Pconnect()
mysql_db_result()
mysql_Num_Rows()</td>
<td>
<pre>mysql_Pconnect(“”, “warenhaus“, “test“);
$query = “select * from artikel“;
$result = mysql_db_query(“warenhaus“, $query);
if (mysql_Num_Rows($result) > 0)
{
   ... weitere Anweisungen ...
}</pre>
mysql_Pconnect() öffnet eine Verbindung zur Datenbank Warenhaus unter Verwendung des Passwortes “test“.

Der Variablen $query wird ein SQL-String zugewiesen.

Mysql_db_result() übermittelt den SQL-String an die Datenbank und bei Erfolg die Ergebnismenge zurück, bei Mißerfolg den Wert “false“. Die Funktion mysql_Num_Rows() prüft, ob die Ergebnismenge Datensätze enthält.
</td>
</tr>
<tr>
<td>mysql_Result()</td>
<td>
<pre>$i = 0;
while ($i < mysql_NumRows($result))
{
   $nr = mysql_Result($result, $i, “nr“);
   $i++;
}</pre>
In einer While-Schleife werden alle Datensätze abgearbeitet.

Mysql_Result() extrahiert aus der Ergebnismenge, die durch “$result“ repräsentiert wird den Inhalt des Feldes “nr“ aus dem Datensatz mit der Nummer “$i“.
</td>
</tr>
</table>

1.3.4.1. Beispielprogramm: Datenbank abfragen (bsp0.php3)

Das folgende Programm stellt eine Verbindung zur Warenhaus-Datenbank her. Desweiteren werden alle Kundenstammdaten aus der Tabelle kunden, sortiert nach Kundennummer ausgelesen.

```
<html>
<head>
<title>Warenhaus Test - Ausgabe aller Kundendaten -</title>
</head>

<body>

<?php
     // Datenbankverbindung herstellen:
    mySql_PConnect( "",  "user",  "user");

     // SQL-Abfrage:
    $query =  "select * from kunden order by kundennummer";
    $result = mysql_db_query( "warenhaus", $query);

     // Überprüfen, ob SQL-Ergebnis nicht leer:
    if (mysql_Num_Rows($result) > 0)
    {
        $Meldung =  "";
    }
    else
    {
        $Meldung =  "Fehler bei der Abfrage oder keine Daten";
    }

     // Wenn Fehler besteht, dann gib Fehlermeldung aus:
    if ($Meldung !=  '')
    {
        printf( "%s", $Meldung);
    }
     // Ansonsten gib gewünschte Daten aus Kundentabelle
    else
    {
        printf( "<div align=center>\n");
```

```
        printf( "<table border=1 >\n");
        printf( "<tr>\n");
            printf( "<th>Kunden-Nr</th>\n");
            printf( "<th>Nachname</th>\n");
            printf( "<th>Vorname</th>\n");
            printf( "<th>Straße</th>\n");
            printf( "<th>Plz</th>\n");
            printf( "<th>Ort</th>\n");
        printf( "</tr>\n");

        $i = 0;
        while ($i < mysql_NumRows($result))
        {
             // Auslesen der Daten aus der Kundentabelle
            $kundennummer = mysql_Result($result, $i,  "kundennummer");
            $nachname     = mysql_Result($result, $i,  "nachname");
            $vorname      = mysql_Result($result, $i,  "vorname");
            $strasse      = mysql_Result($result, $i,  "strasse");
            $plz          = mysql_Result($result, $i,  "plz");
            $wohnort      = mysql_Result($result, $i,  "wohnort");

            printf( "<tr>\n");
                  printf( "<td>%s</td>\n", $kundennummer);
                  printf( "<td>%s</td>\n", $nachname);
                  printf( "<td>%s</td>\n", $vorname);
                  printf( "<td>%s</td>\n", $strasse);
                  printf( "<td>%s</td>\n", $plz);
                  printf( "<td>%s</td>\n", $wohnort);
            printf( "</tr>");
            $i++;
        }
        printf( "</table>\n");
    }
?>

</body>

</html>
```

2. Beispieldatenbank Warenhaus

2.1. Tabellenstruktur und Aufbau des Warenhauses

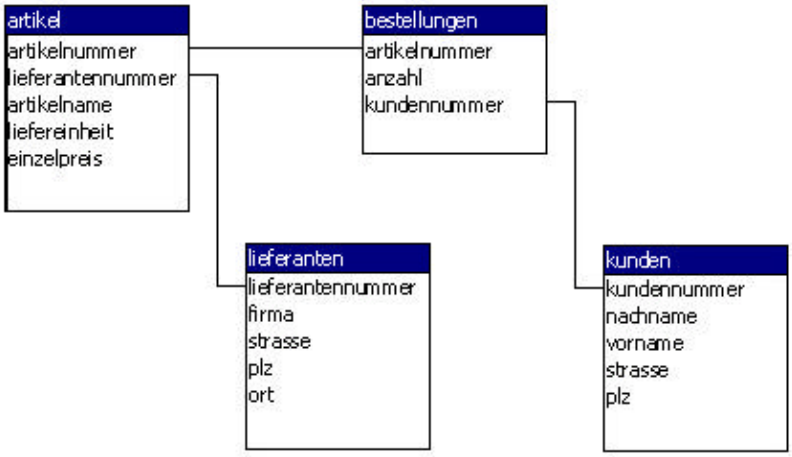


Abbildung 4: Aufbau der Datenbank Warenhaus

Die Warenhaus-Datenbank besteht aus vier Tabellen (vgl. Abbildung 4):

- **Artikel**
  - In dieser Tabelle befinden sich alle Artikelstammdaten.
- **Bestellungen**
  - In der Tabelle bestellungen sind alle Bestellungen abgelegt.
- **Kunden**
  - In der Tabelle kunden befinden sich alle Kundenstammdaten.
- **Lieferanten**
  - In der Tabelle lieferanten befinden sich alle Lieferantenstammdaten.

2.1.1. Tabelle Artikel:

Feld	**Typ**	**Schlüssel**
Artikelnummer	Int(11)	Primärschlüssel
Lieferantennummer	Int(11)	
Artikelname	Char(100)	
Liefereinheit	Char(50)	
Liefereinheit	Char(50)	
Einzelpreis	Double(16,2)	

2.1.2. Tabelle Bestellungen:

Feld	**Typ**	**Schlüssel**
Artikelnummer	Int(11)	Primärschlüssel
Anzahl	Int(11)	
Kundennummer	Int(11)	

2.1.3. Tabelle Kunden:

Feld	**Typ**	**Schlüssel**
Kundennummer	Int(11)	Primärschlüssel
Nachname	Char(30)	
Vorname	Char(30)	
Strasse	Char(30)	
Plz	Char(6)	
Wohnort	Char(30)	

2.1.4. Tabelle Lieferanten :

Feld	**Typ**	**Schlüssel**
Lieferantennummer	Int(11)	Primärschlüssel
Strasse	Char(30)	
Plz	Char(30)	
Ort	Char(30)	

2.2. Rechtevergabe

Um auf die Datenbank zugreifen zu können, müssen die Benutzerrechte, Datenbankname und Hostname in die **mysql**-Datenbank eingetragen werden.

- Benutzername = user
- Benutzerpasswort = user
- Datenbankname = warenhaus
- Hostname = localhost

Bei der Rechtevergabe ist zu bemerken, dass standardmässig alle Rechte auf 'N' gesetzt sind. Diese Werte sind gegebenenfalls anzupassen.

2.3. Darstellung im Internet

Das Warenhaus ist unter folgender Adresse im Internet zu finden:

http://swlablinux.htw-saarland.de/warenhaus/

2.3.1. Überblick

Die folgenden Abbildungen zeigen die Darstellung des Warenhauses mit dem Internet Explorer 5.0. Die Anwendung ist mit Rahmen (Frames) realisiert. Die im unteren Rahmen dargestellten „Karteikarten“ dienen zur Navigation innerhalb der Web-Anwendung. Mit den Links innerhalb der einzelnen Rubriken werden die PHP3-Dateien aufgerufen, die die Ergebnisse in Tabellenform anzeigen.

Beispiel für die Navigation innerhalb der Web-Anwendung:

Beim Aufruf der URL: http://swlablinux.htw-saarland.de/warenhaus/ erscheint die Starteite des Warenhauses (Abbildung 5). Ein User möchte alle Artikel eines bestimmten Lieferanten sehen. Durch Klicken auf den Link „Artikel“ erscheint ein Auswahlmenue (Abbildung 6) mit verschiedenen Abfragemöglichkeiten. Um alle Artikel eines bestimmten Lieferanten anzuzeigen, wird der Link „Alle Artikel eines Lieferanten anzeigen“ angeklickt.

Anschließend erscheint eine Seite (Abbildung 7) mit einer Auswahlbox, die alle in der Datenbank erfassten Lieferantennummern enthält. Der User wählt eine Lieferantennummer aus und bestätigt die Abfrage, indem er mit der Maus auf die Schaltfläche „Anfrage senden“ drückt.

Abschließend erscheint das Abfrageergebnis in einer Tabelle, in der alle Artikel aufgelistet sind (Abbildung 8).

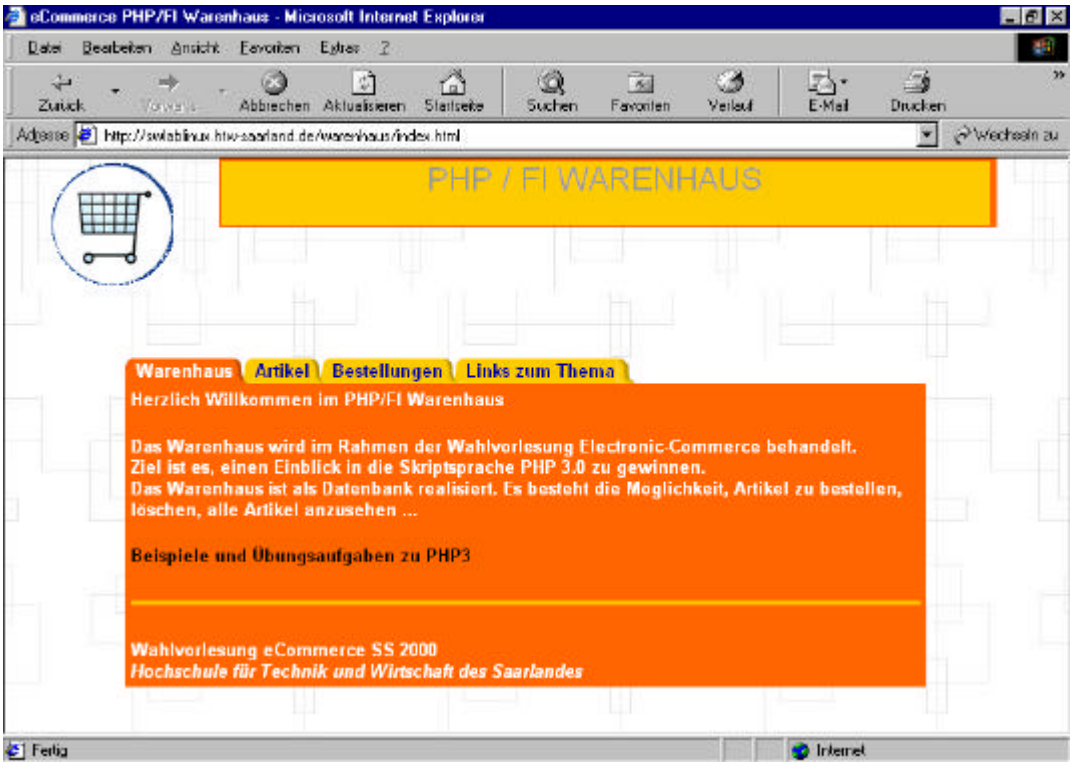


Abbildung 5: Aufbau der Startseite des Warenhauses.

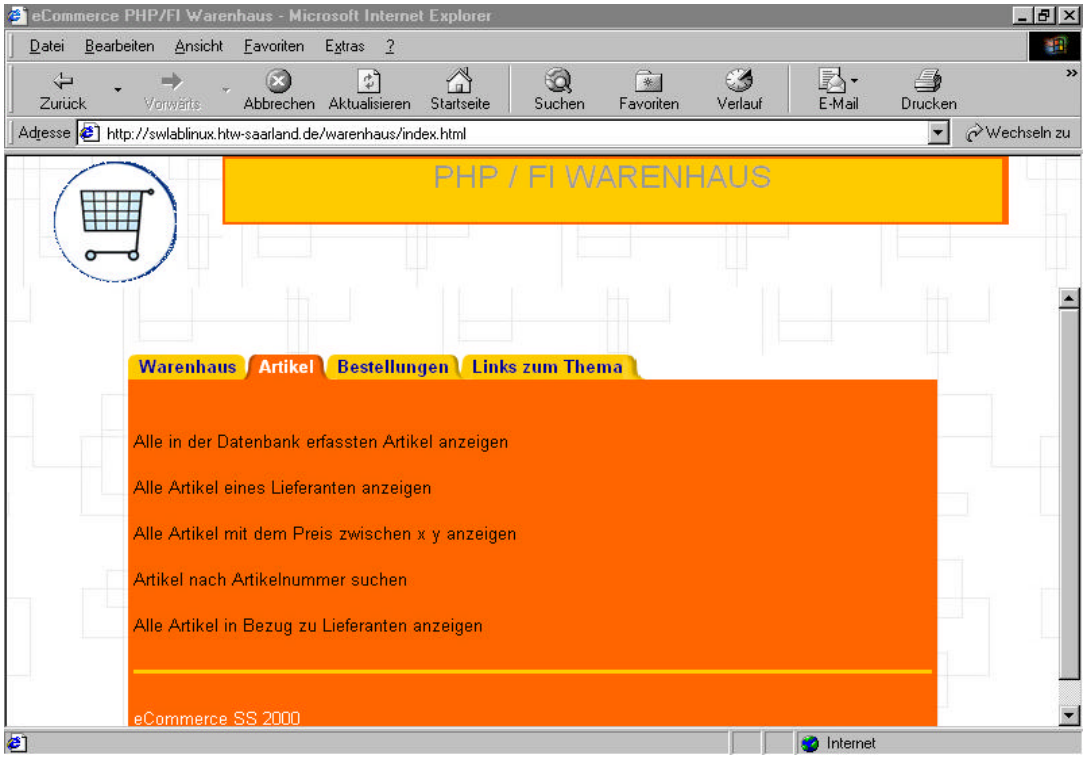


Abbildung 6: Anklicken von „Alle Artikel eines Lieferanten“ im Bereich „Artikel“.

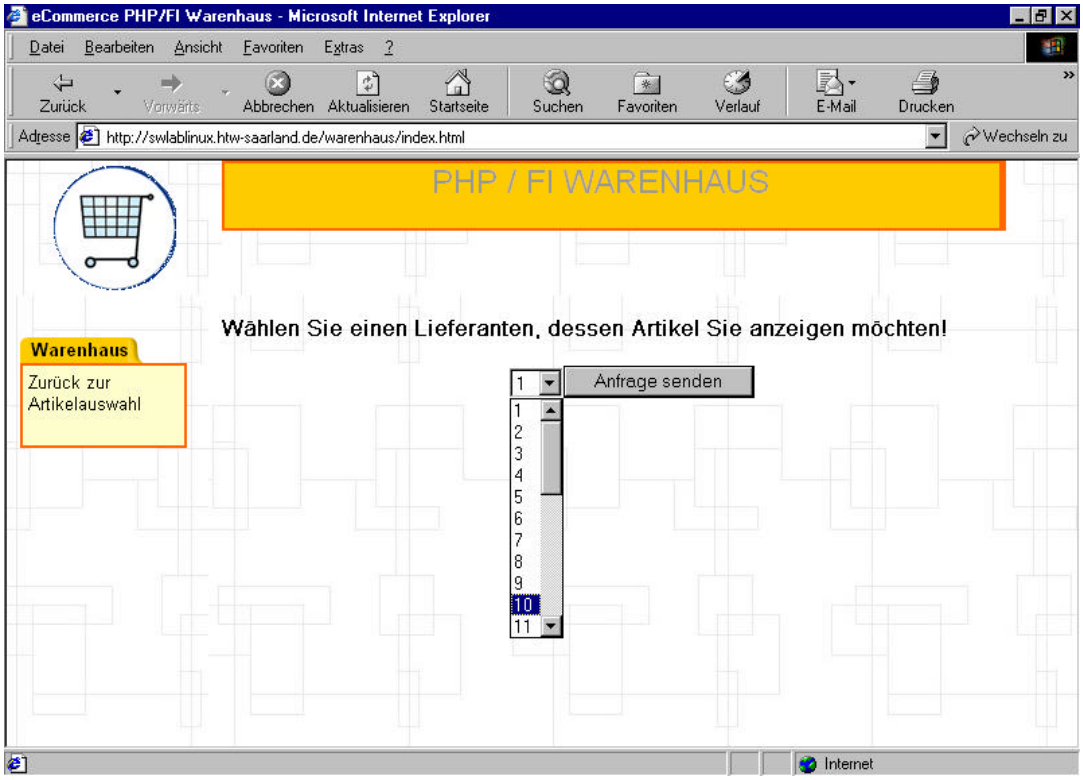


Abbildung 7: Auswahl eines Lieferanten, dessen Artikel angezeigt werden sollen.

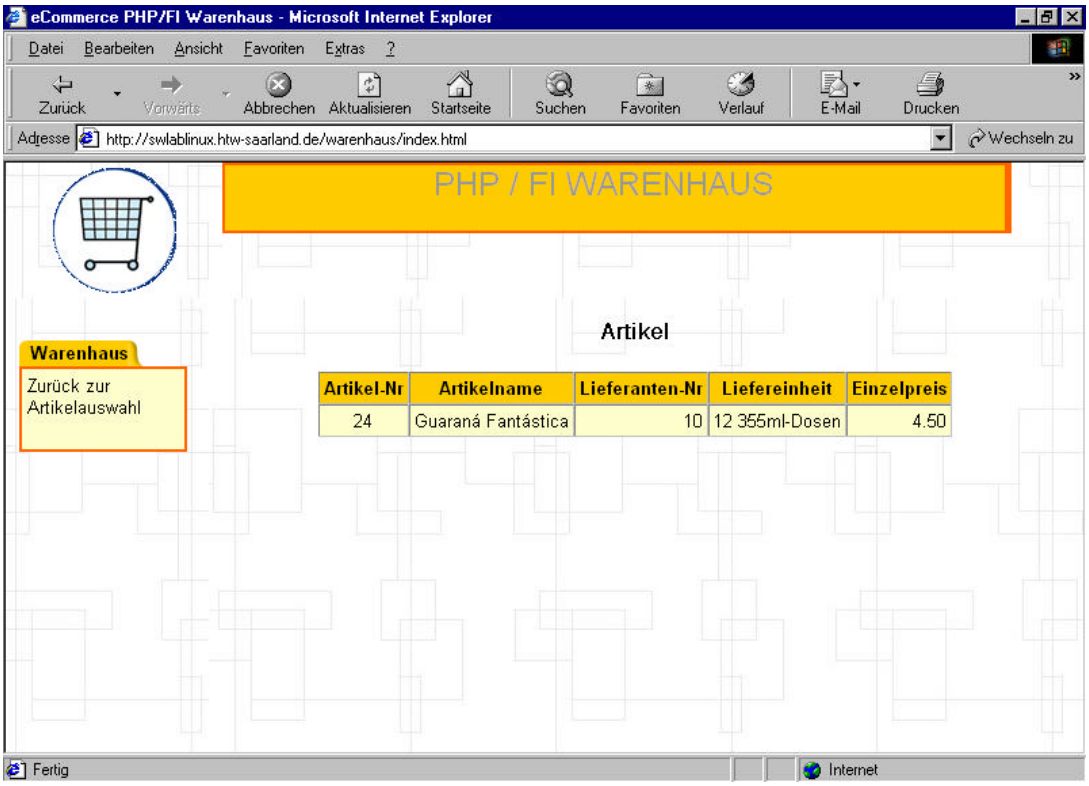


Abbildung 8: Ergebnistabelle, nach der Suche aller Artikel eines bestimmten Lieferanten.

3. Übungsaufgaben

3.1. *Artikel*

3.1.1. Alle Artikel anzeigen

Aufgabenstellung:

In dieser Aufgabe sollen alle Artikeldaten aus der Tabelle **artikel** sortiert nach Artikelnummer in Tabellenform ausgegeben werden. Die Tabelle enthält folgende Spalten:

- Artikelnummer,
- Artikelname,
- Lieferantennummer,
- Liefereinheit,
- Einzelpreis.

3.1.2. Artikel eines Lieferanten

Aufgabenstellung:

In einem HTML-Formular soll eine Optionsliste mit allen Lieferantennummern gefüllt werden. Ein User soll die Möglichkeit haben, einen Lieferanten auszuwählen. Über eine geeignete Schaltfläche sollen anschließend alle Artikel aufgelistet werden, die von diesem Lieferanten bezogen werden können.

3.1.3. Artikeleinschränkung nach Preisen

Aufgabenstellung:

In einem HTML-Formular mit zwei Eingabefeldern sollen zwei DM-Werte erfasst werden (Preis von - bis). Die Ergebnistabelle soll alle Artikel enthalten, deren Einzelpreis zwischen den eingegebenen Preisen liegt (einschließlich der Eingabewerte). Sortiert wird absteigend nach dem Feld Einzelpreis.

3.1.4. Suche eines Artikels

Aufgabenstellung:

In einem HTML-Formular mit einem Eingabefeld wird ein Suchstring erfasst. Die Suche soll im Feld Artikelname erfolgen. Es sollen auch Teilstrings gefunden werden.

Beispiel: Die Eingabe „ff“ findet „Kaffee“.

3.1.5. Gegenüberstellung Artikel - Lieferant

Aufgabenstellung:

In einer Tabelle sollen alle Lieferanten mit ihrem Warensortiment aufgelistet werden.

3.1.6. Bestellung eines Artikels

Aufgabenstellung:

In einem HTML-Formular soll ein User die Möglichkeit haben, unter Angabe von Kundennummer, Artikelnummer und Menge einen Artikel zu bestellen.

Lösungshinweis: INSERT-Anweisung.

3.2 Bestellungen

3.2.1. Ausgabe aller Bestellungen

Aufgabenstellung:

Ausgabe einer Tabelle aller Bestellungen.

Auszugebende Felder:

- Kundennummer,
- Artikelnummer,
- Artikelname,
- Anzahl,
- Einzelpreis,
- Gesamtpreis.

Sortierung nach Kundennummer und Artikelnummer.

3.2.2. Löschen von Bestellungen

Aufgabenstellung:

In einer Tabelle soll die Möglichkeit bestehen, einzelne Bestellungen zu löschen.

Auszugebende Felder:

- Artikelnummer,
- Artikelname,
- Kundennummer,
- Anzahl,
- Einzelpreis,
- Gesamtpreis.

4. Zugriff auf Dateisystem

Der Linux-Server ist so eingerichtet, dass von WindowsNT auf das Linux-Dateisytem zugegriffen werden kann. Um auf das Dateisystem von Linux zuzugreifen, stellt der Übungsteilnehmer eine Netzwerkverbindung von WindowsNT zu Linux her. Zur Anmeldung muß der Übungsteilnehmer unter der Kennung PI3 an einer WindowsNT-Workstation angemeldet sein.

Vorgehensweise zur Netzwerkverbindung:

- Netzlaufwerk verbinden (Abbildung 9)
  \\swlablinux\ecommerce\

Alle Übungen werden unter diesem Verzeichnis abgelegt. Es ist sinnvoll zur Bearbeitung entsprechnende Unterverzeichnisse anzulegen.

Das Warenhaus befindet sich unter folgender Adresse:
http://swlablinux.htw-saarland.de/warenhaus/

Das Verzeichnis ecommerce befindet sich unter folgender Adresse:
http://swlablinux.htw-saarland.de/ecommerce/

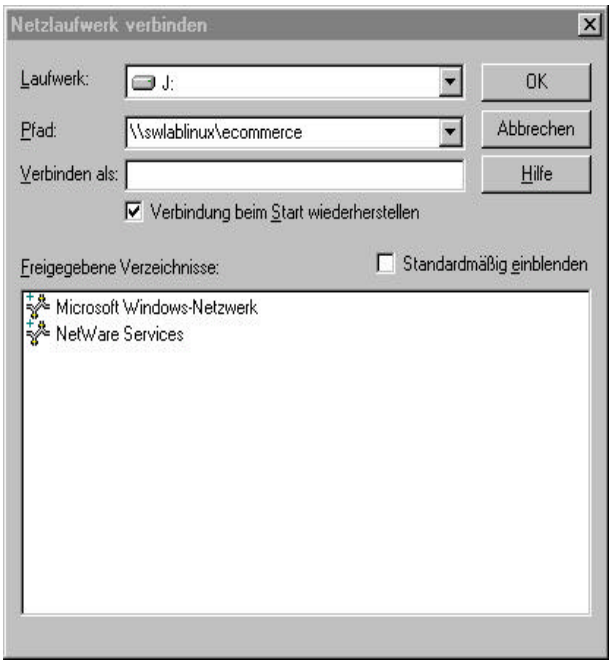


Abbildung 9: Herstellen der Netzlaufwerk-Verbindung zu SWLABLINUX

5. Quellennachweis:

5.1. Bücher

PHP dynamische Webauftritte professionell realisieren
Egon Schmid, Christian Cartus, Richard Blume
Verlag: Markt&Technik
ISBN: 3-8272-5524
URL: http://www.mut.de

PHP – Grundlagen und Lösungen –
Webserver-Programmierung unter Windows und Linux
Jörg Krause
Verlag: Hanser
ISBN: 3-446-21301-5
URL: http://www.hanser.de

MySQL & msql – Databases for Moderate-Sized Organizations & Web Sites
Randy Jay Yarger, George Reese & Tim King
Verlag: O'Reilly
ISBN 1-56592-434-7
URL: http://www.oreilly.com

5.2 Links zu PHP/MySQL

5.2.1. Dokumentationen zu PHP und MySQL

http://www.php.net
http://www.php3.com
http://www.php-center.de
http://www.php-area.de/
http://www.phpbuilder.com/
http://www.mysql.com

5.2.2. Mailinglisten

http://infosoc.uni-koeln.de/php/suche/
http://infosoc.uni-koeln.de/mailman/listinfo/php
http://www.elysium.pl/members/brush/php-ezmlm/index.html
http://www.devshed.com/Talk/Mailing_List/
http://www.analogon.com/php/forum/index.php3

5.2.3. MySQL Administrationsprogramme

http://www.phpwizard.net/
http://www.314interactive.com/software/map/
http://www.judas-price.de/
http://www.phphelp.com/
http://modems.rosenet.net/mysql/
http://myadmin.cheapnet.net/