Serienbriefe mit LaTeX und MySQL

Uwe Ziegenhagen
www.ziegenhagen.info

22. Februar 2003

Dieser Artikel soll zeigen wie man mittels MySQL und PHP Serienbriefe für LaTeX gestalten kann. Es gibt sicher noch bessere Möglichkeiten der Integration, aus Zeitmangel habe mich aber damals für exakt diesen Weg entschieden.
Zu meiner Person: Ich studiere BWL an der Humboldt-Universität zu Berlin mit Schwerpunkt Wirtschaftsinformatik, Entrepreneurship und Statistik. Von 2000-2002 habe ich an meinem Institut die Organisation der Compstat 2002-Konferenz übernommen. Im Rahmen dieser Konferenz mußten verschiedene personalisierte Dokumente und Tagungsunterlagen erstellt werden.

1 Benötigte Hard- und Software

Die Erstellung der LaTeX-Dokumente erfolgte mit MikTeX 2 auf einem Win2000-Rechner. Als Plattform für MySQL diente der gleiche Rechner mit

1. einem lokalen Apache-Webserver
2. einer installierten Windowsversion von MySQL
3. PHP4, das Unterstützung für MySQL eingebaut hat
4. phpMyAdmin, ein PHP-basiertes Administrationstool für MySQL-Datenbanken, das eine Menge Arbeit und Ärger erspart

Bei jeder guten Linux-Distribution wird diese Software schon mitgeliefert, als MS-Yunkie muss man sich etwas in die Konfiguration reinknien. Bei Fragen zu PHP schaue man zuerst in die FAQ von de.comp.lang.php oder in die Newsgroup de.comp.lang.php.misc.

Ein sehr gutes Buch zu PHP ist das von Jörg Krause, *PHP 4 - Grundlagen und Profiwissen*. Mir hat es sehr geholfen, insbesondere das Kapitel zu regulären Ausdrücken ist sehr umfangreich.

2 Erstes Beispiel

2.1 Datenbank-Struktur

Beginnen wir mit einem kleinen Beispiel: Wir haben eine Datenbank mit nur einer Tabelle, die ein einige Adressen enthält. Diese Adressen wollen wir in einen SCRLETTER2-Serienbrief übernehmen. Folgend die benötigten SQL-Befehle, um diese Tabelle (unter der Annahme, dass die Datenbank schon formal erstellt) zu erstellen. Diese Befehle kann man im phpMyAdmin in das SQL-Feld eingeben oder per Hand einladen (Achtung: Der Zeilenumbruch bei den Daten erfolgt hier nur wegen der besseren Lesbarkeit.

```
# Tabellenstruktur für Tabelle ‘adr‘
#

CREATE TABLE adr (
  Vorname varchar(15) NOT NULL default ’’,
  Name varchar(35) NOT NULL default ’’,
  Strasse varchar(40) NOT NULL default ’’,
  PLZ varchar(10) NOT NULL default ’’,
  Ort varchar(30) NOT NULL default ’’
) TYPE=MyISAM;

#
# Daten für Tabelle ‘adr‘
#

INSERT INTO adr VALUES (’Harry’, ’Hirsch’, ’Hirschweg 5’, ’12345’, ’Berlin’);
INSERT INTO adr VALUES (’Donald’, ’Duck’, ’Erpelgasse 1’,
’23451’, ’Entenhausen’);
INSERT INTO adr VALUES (’Hans’, ’Mustermann’, ’Musterstrasse 26a’,
’127596A’, ’Neubrandenburg’);
INSERT INTO adr VALUES (’Dagobert’, ’Duck’, ’Talerallee 25’,
’102012’, ’Entenhausen’);
INSERT INTO adr VALUES (’Daisy’, ’Duck’, ’Ganterweg 12a’,
’564879’, ’Erpelhausen’);
```

2.2 Zugriff auf die Datenbank

Wenn die Daten in der Datenbank verfügbar sind, wird es Zeit zu prüfen, ob wir auch darauf zugreifen können. Mit dem folgenden PHP-Skript verbinden wir uns mit der Datenbank und fragen, wieviele Einträge vorhanden sind. Die Datei wird über den Browser aufgerufen.

```
1 <?php
2         $link = mysql_connect("localhost", "sys", "sys");
3         // Verbindung mit Rechneradresse, login und passwort
4         mysql_select_db("TeX-1"); // Auswahl der Datenbank
5         $query = "SELECT count(*) FROM adr"; // Abfrage
```

```
6         $result = mysql_query($query); // das Ergebnis
7         while ($line = mysql_fetch_array($result, MYSQL_ASSOC))
8         {
9                 foreach ($line as $col_value) {
10          print "<b>$col_value</b><br>";
11
12
13                                  }
14                          }
15         mysql_free_result($result);
16         mysql_close($link); // Beende Verbindung.
17 ?>
```

Wenn nirgendwo ein Tippfehler drin ist und die Zugangsdaten richtig sind, wird **5** als Ergebnis angezeigt. Als nächstes lassen wir uns mal die Adressen komplett im Browser anzeigen (schön als Tabelle formatiert):

```
1  <?php
2         $link = mysql_connect("localhost", "sys", "sys");
3         mysql_select_db("TeX-1");
4         $query = "SELECT * FROM adr";
5         // Waehle alles aus!
6         $result = mysql_query($query);
7         print "<table border=\"1\">";
8         while ($line = mysql_fetch_array($result, MYSQL_ASSOC))
9         {
10          print "<tr>";
11          foreach ($line as $col_value) {
12          print "<TD>$col_value</TD> ";
13
14                                  }
15         print "</tr>";
16                  }
17
18                  print "</table>";
19         mysql_free_result($result);
20         mysql_close($link);
21 ?>
```

Wenn alles geklappt hat, sollte es im Webbrowser so aussehen:

Harry	Hirsch	Hirschweg 5	12345	Berlin
Donald	Duck	Erpelgasse 1	23451	Entenhausen
Hans	Mustermann	Musterstrasse 26a	127596A	Neubrandenburg
Dagobert	Duck	Talerallee 25	102012	Entenhausen
Daisy	Duck	Ganterweg 12a	564879	Erpelhausen

2.3 Die Verbindung zu LaTeX

So, nun kümmern wir uns mal um die Verbindung zu LaTeX. Folgend ein Musterbrief, erstellt mit der SCRLETTER2-Klasse aus dem Koma-Paket. Es fällt auf, dass wir für einen Serienbrief nur das Stück zwischen \begin{letter} und \end {letter} abändern müssen. Das bedeutet für uns, den Kopf und das Ende des Dokuments lassen wir als Reihe von Strings durch PHP ausgeben, für die individuellen Briefe nehmen wir eine Schleife.

```
\documentclass[DIN,fromalign=center,fromfax=true,fromphone=true]{scrlttr2}
\usepackage[german]{babel}
\usepackage[latin1]{inputenc}
\setkomavar{fromfax}{030-123-4567}
\setkomavar{fromphone}{030-123-4568}
\setkomavar{fromname}{Hans Mustermann}
\setkomavar{fromaddress}{Muhweg 1 12345 Muhdorf}
\setkomavar{subject}{Werbung}
\setkomavar{yourmail}{02.09.2002}
\setkomavar{yourref}{müller2002}
\begin{document}
\begin{letter}{Eva Adam \\ Musterweg 2\\12345 Muskau}
\opening{Sehr geehrte Eva Adam,}

anbei schicke ich Ihnen sinnlose Werbung.

\closing{Mit freundlichen Grüßen}
\end{letter}
```

```
1  <?php
2  echo "\\documentclass[DIN,fromalign=center,fromfax=true,";
3  echo "fromphone=true]{scrlttr2}<br>";
4  echo "\\usepackage[german]{babel}<br>";
5  echo "\\usepackage[latin1]{inputenc}<br>";
6  echo "\\setkomavar{fromfax}{030-123-4567}<br>";
7  echo "\\setkomavar{fromphone}{030-123-4568}<br>";
8  echo "\\setkomavar{fromname}{Hans Mustermann}<br>";
9  echo "\\setkomavar{fromaddress}{Musterweg 1 12345 Musterstadt}<br>";
10 echo "\\setkomavar{subject}{Werbung}<br>";
11 echo "\\setkomavar{yourmail}{02.09.2002}<br>";
12 echo "\\setkomavar{yourref}{ümller2002}<br>";
13 echo "\\begin{document}<br>";
14
15
16          $link = mysql_connect("localhost", "sys", "sys");
17          mysql_select_db("TeX-1");
18          $query = "SELECT * FROM adr";
19          $result = mysql_query($query);
20          $n=mysql_num_rows($result)-1;
21
22          for ($i = 0; $i <= $n; $i++) {
```

```
23          echo "\\begin{letter}";
24          echo "{".mysql_result($result,$i,"Vorname")." ";
25          echo mysql_result($result,$i,"Name")." ";
26          echo "\\\\ ".mysql_result($result,$i,"Strasse")." \\\\ ";
27          echo mysql_result($result,$i,"PLZ");
28          echo " ".mysql_result($result,$i,"Ort")."}";
29
30
31          echo "\\opening{Sehr geehrte(r) ";
32          echo mysql_result($result,$i,"Vorname");
33          echo " ".mysql_result($result,$i,"Name").",}<br>";
34          echo "anbei schicke ich Ihnen sinnlose Werbung.<br>";
35          echo "\\closing{Mit freundlichen üßGren}<br>";
36          echo "\\end{letter}<br>";
37  }
38
39  echo "\\end{document}";
40
41  mysql_free_result($result);
42  mysql_close($link);
43  ?>
```

Wir unterscheiden hier nicht zwischen Männlein und Weiblein, dies wäre aber mit einem zusätzlichen Feld in der Datenbank und einer `if`-Abfrage im PHP-Skript leicht zu realisieren.

3 Ein komplexes Beispiel

Jetzt zu einem komplexen Beispiel: Für die Konferenz mußten wir ein Konferenzprogramm erstellen, dem die Teilnehmer entnehmen konnten, wo welcher Vortrag stattfinden würde. Die gesamten Daten der Tagung lagen auf einem Sybase-Datenbankserver, der über ein Webinterface bedient werden konnte[1]. Per SQL-Abfrage wurden dann aus der Datenbank die Daten extrahiert, die im Tagungsprogramm benötigt wurden und anschließend über den Umweg eines Textfiles in MySQL importiert[2].

Ich hatte dann die folgende Tabelle in MySQL:

- Tag war der Tag, logisch
- startsess die Anfangszeit der Vortragsgruppe
- endsess die Endzeit
- raum der Ort
- track der Name der Vortragsgruppe (Dataming, Algorithms)

[1] Wer sich von den Lesern dieses Artikels für die Einzelheiten interessiert (weil vielleicht selbst eine Konferenz zu managen ist), möge sich bei mir melden, dieses System war ein fantastisches Projekt unseres Rechenzentrums

[2] Man hätte natürlich auch ohne MySQL auf Sybase arbeiten können, übers Modem macht es sich aber schwer.

- session der Name der Session (Datamining1, Datamining1, etc.)
- vorchair der Vorname des Leiters der Session
- nachchair der Nachname
- paper der Titel des einzelnen Papers
- voraut der Vorname des Vortragenden
- nachaut der Nachname

Die Tabellenstruktur sah wie folgt aus, es gab 178 Einträge (da es 178 Vorträge gab):

```
CREATE TABLE ‘compstat‘.‘schedule5‘ (
  ‘tag‘ varchar(12) NOT NULL default ’’,
  ‘startsess‘ varchar(5) NOT NULL default ’’,
  ‘endsess‘ varchar(5) NOT NULL default ’’,
  ‘raum‘ varchar(12) NOT NULL default ’’,
  ‘track‘ varchar(25) NOT NULL default ’’,
  ‘session‘ varchar(25) NOT NULL default ’’,
  ‘vorchair‘ varchar(15) NOT NULL default ’’,
  ‘nachchair‘ varchar(15) NOT NULL default ’’,
  ‘starttalk‘ varchar(5) NOT NULL default ’’,
  ‘paper‘ text NOT NULL,
  ‘voraut‘ varchar(15) NOT NULL default ’’,
  ‘nachaut‘ varchar(15) NOT NULL default ’’
) TYPE=MyISAM
```

Hauptproblem war nun, aus dieser einen Tabelle das folgende Layout für jeden einzelnen Tag zu erzeugen.

Aug 24 2002

13:30-14:15 Invited Session 1 (3094) Chair: Michael G. Schimek

13:30 An implementation for regression quantile estimation
Thomas Yee

13:30-15:00 Robust 1 (2097) Chair: Jaromir Antoch

13:30 A Hotelling Test based on MCD
Gert Willems

14:00 Data Depth and Quality Control
Giovanni C. Porzio

14:15 Experiments of robust ESACF identification of ARIMA models
Heikki Hella

14:30 Analyzing data with robust multivariate methods and diagnostic plots
Greet Pison

Man sieht, zuerst mußten die Einträge nach dem Tag sortiert werden, dann in zeitlich aufsteigender Reihenfolge. Name der Session, Ort und Leiter mußten stimmen, anschließend mußten die einzelnen Vorträge in korrekter Reihenfolge mit den richtigen Daten ausgegeben werden.

Diese Arbeit erledigt ein PHP-Skript von knapp 50 Zeilen (das mich mehrere Tage meines Lebens und meiner geistigen Verfassung :-) gekostet hat). Da der Quellcode recht breit ist verzichte ich hier auf das Einfügen als Listing und lege ihn als Textdatei in das Zip-Archiv dieses Artikels, außerdem steht er unter den \end{ document } des Quelltextes dieses Dokuments.

3.1 Was passiert in diesem Skript?

Zuerst regeln wir die Ausgabe der Präambel unseres LaTeX-Dokuments und nehmen anschließend Kontakt zum MySQL-Server auf. Mit der ersten `distinct`-Abfrage holen wir uns die unterschiedlichen Tage in ein MySQL-result.

Für jeden einzelnen Tag geht's dann weiter: Erst geben wir den Tag aus (als HTML formatiert, erleichert die Fehlerkorrektur und ist unsichtbar im LaTeX-Code) und basteln eine Abfrage und ein dazugehöriges `result2`, wo wir die unterschiedlichen Sessions an **diesem** Tag erfassen.

Die Session wird ausgegeben bevor wir uns der finalen Abfrage mit dem Ergebnis `result3` widmen: Wir wollen alle Werte der Tabelle an **diesem** Tag und **dieser** Session.

Diese Abfrage wird aus den Ergebnissen der beiden Abfragen `query` und `query2` zusammengebaut und als `query3` an den MySQL-Server geschickt.

Wir holen uns die Zahl der passenden Werte in die Variable n und lassen dann eine `for`-Schleife laufen, in der die folgenden trickreichen Dinge geschehen (mehrmals lesen): Wenn die Variable `$session` schon mit dem Wert der aktuellen Session belegt ist, brauchen wir die Daten der Session nicht mehr ausgeben. Falls die Variable **nicht** diesen Wert hat, wird er ihr zugewiesen und die Daten der Sessions werden als HTML ausgegeben.

Die Zeile `$paper=...` tut nichts anderes als die ampersands zu escapen (gutes Deutsch), was für unsere anschließende Ausgabe der Daten als Tabelle wichtig ist (generell sollte man alle Zeichen filtern, die LaTeXals Teil eines Befehls kennt).

Die nächsten drei `echo`-Zeilen geben dann die Daten der einzelnen Vorträge als Tabelle aus, zuletzt schließen wir die Verbindungen zu MySQL und übertragen per Copy&Paste den LaTeX-Quelltext in einen Texteditor.

Auf die gleiche Weise wurde dann noch ein Serienbrief generiert, in dem die einzelnen Leiter der Sessions sehen konnten, wann sie wo zu sein hatten. Da das Vorgehen ziemlich ähnlich zu dem letzten Beispiel war, lasse ich es hier mal weg.

Nachwort

Ich hoffe, ich habe euch ein paar Möglichkeiten gezeigt, mit PHP und MySQL Daten für LaTeXu formatieren, über Vorschläge und Anregungen würde ich mich freuen.