Datenbanken mit MySQL

In diesem Abschnitt werden die Grundlagen des Aufbaus und der Struktur von Datenbanken anhand des MySQL-Datenbank-Servers beschrieben. Im Zusammenhang mit PHP werden sehr häufig MySQL-Datenbanken verwendet.

Datenbank, Tabelle

Eine Datenbank dient zur Speicherung größerer Datenmengen und zur übersichtlichen Darstellung bestimmter Daten aus diesen Datenmengen. Innerhalb einer Datenbank befinden sich verschiedene Tabellen, hier eine einfache Beispiel-Tabelle:

Name	Vorname	Personalnummer	Gehalt	Geburtstag
Maier	Hans	6714	3500,00	15.03.62
Schmitz	Peter	81343	3750,00	12.04.58
Mertens	Julia	2297	3621,50	30.12.59

Feld, Datensatz

Die Begriffe in der ersten Zeile nennt man die Datenfelder der Tabelle. Anschließend folgen die einzelnen Datensätze der Tabelle, in diesem Falle drei.

Struktur, Daten

Natürlich legt niemand für drei Datensätze eine Datenbank mit einer Tabelle an, aber die vorliegende Struktur könnte auch für mehrere tausend Datensätze verwendet werden. Die Datenfelder haben einen bestimmten Datentyp, hier sind dies Text, Zahlen und Datumsangaben.

Beim Erzeugen einer Datenbank geht man wie folgt vor:

- Anlegen der Datenbank
- Anlegen von Tabellen durch Angabe der Struktur
- Eingeben der Datensätze in die Tabellen

Struktur-Änderung

Die Struktur einer existierenden Datenbank bzw. einer Tabelle kann auch noch verändert werden, wenn sich bereits Daten darin befinden. Allerdings ist es empfehlenswert, sich vorher gründlich Gedanken über die Struktur einer Datenbank zu machen, da bei einer nachträglichen Veränderung leicht Datenverluste auftreten können.

1 MySQL: Installation und Start

1.1 Datenbanken mit MySQL

SQL, MySQL

Im Zusammenhang mit der Programmiersprache PHP wird häufig mit MySQL-Datenbanken gearbeitet. MySQL ist ein SQL-basierter Datenbank-Server. SQL ist die meistverwendete Datenbanksprache der Welt. MySQL bietet SQL-Anweisungen

- zur Erzeugung der Struktur von Datenbanken und Tabellen,
- zum Bearbeiten der Datensätze (Erzeugen, Anzeigen, Ändern, Löschen).

MySQL und PHP

Der Entwickler einer Datenbank legt normalerweise die Struktur mittels MySQL an und stellt dem Benutzer zur komfortablen Bearbeitung der Datensätze HTML-Dateien mit PHP-Programmen zur Verfügung.

Dies ist auch in den folgenden Abschnitten berücksichtigt worden:

- Zunächst werden die wichtigsten SQL-Anweisungen zur Benutzung von MySQL und zur Erzeugung bzw. Veränderung der Struktur einer Datenbank mit MySQL beschrieben.
- Es folgen die SQL-Anweisungen zum Erzeugen, Anzeigen, Ändern und Löschen von Datensätzen mit MySQL.
- Diese Anweisungen werden dann später (in sehr ähnlicher Form) in den PHP-Programmen zur komfortablen Benutzung der Datenbanken eingesetzt.

MySQL installieren unter Windows

MySQL unter Windows

Bei der Installierung gilt es, zunächst die Datei **mysql-shareware-3.22.33-win.zip** entpacken, anschließend die Datei **setup.exe** aufzurufen und den Installationsanweisungen zu folgen, das Ziel-Verzeichnis heißt **c:\mysql**.

MySQL installieren unter Linux

MySQL unter Linux

Nach der im ersten Abschnitt vorgenommenen Installation des »Suse Systems für Netzwerkserver«, die für die PHP-Programmierung benötigt wird, ist MySQL bereits vorinstalliert.

Die Anweisung mysql_install_db (im Verzeichnis **/usr/bin**) muss einmalig aufgerufen werden. Dadurch werden MySQL-eigene Datenbanken zu Verwaltungszwecken installiert. Falls man versehentlich diese Anweisung später noch einmal aufruft, hat dies keine Nachteile für die bisher erzeugten Datenbanken. Sie werden nicht verändert oder beeinflusst.

MySQL-Server starten und beenden unter Windows

Server starten unter Windows

Nach erfolgter Installation (z. B. in das Verzeichnis **c:\mysql**) geht man in den DOS-Eingabemodus (über Startmenü|Programme|MS-DOS-Eingabeaufforderung). Dort muß man in das Verzeichnis **c:\mysql\bin** wechseln. Hier kann der MySQL-Datenbank-Server durch die Anweisung mysqld-shareware gestartet werden.

Nach der Verwendung sollte man den Datenbank-Server mit der Anweisung mysqladmin -u root shutdown wieder herunterfahren und kann den DOS-Eingabemodus mit Hilfe von exit wieder beenden.

Falls später ein PHP-Programm auf eine vorhandene MySQL-Datenbank zugreift, muss der MySQL-Datenbank-Server (mit mysqld-shareware) vorher gestartet werden. Er sollte anschließend auch beendet werden.

Ein Tipp: Erzeugen Sie mit Hilfe eines Text-Editors die folgenden zwei Dateien:

- die Datei **MySQLStart.bat**, in der die beiden Anweisungen c:\mysql\bin\mysqld-shareware und c:\mysql\bin\mysqlshow stehen
- die Datei **MySQLStop.bat**, in der die Anweisung c:\mysql\bin\mysqladmin -u root shutdown steht

Erstellen Sie je eine Verknüpfung auf die beiden Dateien, und legen Sie diese Verknüpfung auf den Desktop. Auf diese Weise kann der Datenbank-Server einfach mit Doppelklick gestartet und beendet werden.

Zusätzlich werden Ihnen noch beim Start mit Hilfe von mysqlshow die vorhandenen Datenbanken angezeigt. Das Ergebnis für den Aufruf von mysqlshow sieht nach einer Neu-Installation von MySQL wie folgt aus:

```
Databases
mysql
```

MySQL-Server starten und beenden unter Linux

Server starten unter Linux

Zunächst sollte man in das Verzeichnis **/usr/bin** wechseln. Der MySQL-Datenbank-Server kann nun durch die Anweisung safe_mysqld & gestartet werden.

Nach der Verwendung sollte man den Datenbank-Server mit der Anweisung mysqladmin -u root shutdown wieder herunterfahren.

Falls später ein PHP-Programm auf eine vorhandene MySQL-Datenbank zugreift, muss der MySQL-Datenbank-Server (mit safe_mysqld &) vorher gestartet werden. Er sollte anschließend auch beendet werden.

1.2 Der MySQL-Monitor

MySQL-Monitor

Nach erfolgtem Start des MySQL-Datenbank-Servers wechselt man mit der Anweisung mysql in den Monitormodus für MySQL. Es erscheint das MySQL-Prompt. Hier können u. a. die Anweisungen zur Erzeugung von Datenbanken und Tabellen eingegeben werden. Die Eingabe von quit beendet den Monitormodus von MySQL.

Der Start des Monitormodus sieht unter Windows wie folgt aus:

```
c:\mysql> cd bin
c:\mysql\bin> mysql
Welcome to the MySQL-Monitor. Commands end with ; or \g
Your MySQL Connection id is 26 to server version:
3.22.29-shareware-debug
Type 'help' for help
mysql>
```

2 MySQL: Struktur von Datenbank und Tabelle

2.1 Datenbank erzeugen, benutzen, löschen

Nach dem Aufruf des MySQL-Monitors können die nachfolgend an Beispielen gezeigten SQL-Anweisungen zur Datenbankverwaltung eingesetzt werden.

create database

Datenbank erzeugen

```
create database firma;
```

- Die Datenbank mit dem Namen firma wird erzeugt.
- Falls die Datenbank neu erzeugt wurde, so erscheint die Ausgabe Query OK, 1 row affected und eine Zeitangabe.
- Falls sie aber schon existiert und zumindest eine Tabelle besitzt, erfolgt die Fehlermeldung Error 1007: Can't create database 'firma'. Database exists.

use

Datenbank benutzen

```
use firma;
```

- Wechselt auf die Datenbank firma. Dies ist zur weiteren Benutzung dieser Datenbank notwendig.
- Es erscheint die Meldung Database changed.

drop database

Datenbank löschen

```
drop database firma;
```

Löscht die gesamte Datenbank inkl. aller Tabellen und Datensätze. Dieser Befehl sollte nur in Ausnahmefällen angewendet werden!

Datenbanken allgemein

Nachdem die Datenbank firma mit create database erzeugt wurde, kann man später den MySQL-Monitor auch mit mysql firma starten. Die genannte Datenbank ist dann bereits in Benutzung.

Bei den Namen von Datenbanken, Tabellen und Feldern sollte man darauf achten, dass keine deutschen Umlaute, scharfes ß, Leerzeichen und Sonderzeichen verwendet werden.

Zugriffs-Rechte

Innerhalb von MySQL kann mit einem umfangreichen und detaillierten System zur Zugriffssicherung gearbeitet werden. Verschiedenen Benutzern können spezifische Rechte eingeräumt bzw. verwehrt werden. Zum Erlernen des ersten Umgangs mit Datenbanken und der Datenbank-Programmierung wird hier vereinfacht angenommen, daß die Benutzer sämtliche Zugriffsrechte bei der Benutzung von Datenbanken haben.

2.2 Tabelle erzeugen

create table

Nach Erzeugung einer Datenbank können Tabellen angelegt werden. Eine neue Tabelle wird mit Hilfe der SQL-Anweisung create table erzeugt. Ein Beispiel:

```
create table personen
(
   name char(30),
   vorname char(25),
   personalnummer int,
   gehalt double(10,2),
   geburtstag date
);
```

Damit wird eine neue Tabelle mit dem Namen personen erzeugt. Diese Tabelle enthält fünf Felder mit den folgenden Feldnamen und Datentypen:

Feldname	Datentyp	Bedeutung des Datentyps
name	char(30)	Zeichenkette der Länge 30
vorname	char(25)	Zeichenkette der Länge 25
personalnummer	int	ganze Zahlen (im Bereich von –2 147 483 648 bis 2 147 483 647)
gehalt	double(10,2)	Zahlen mit Nachkommastellen, maximale Breite der Anzeige ist 10, die Zahl ist auf zwei Nachkommastellen genau
geburtstag	date	Datumsangaben

Datentypen

Bei diesem Beispiel wurden einige häufig verwendete Datentypen eingesetzt, die für viele Anwendungen bereits ausreichen. Zum Einsatz weiterer Datentypen und Feldeigenschaften wird an dieser Stelle auf das Manual zu MySQL (auf der CD) verwiesen.

2.3 Tabelle komprimieren, löschen

optimize table

Beim Löschen von Datensätzen werden die betroffenen Datensätze nur als »gelöscht« markiert, aber nicht physikalisch gelöscht. Die Tabelle und damit die Datenbank wird durch neue Einträge ständig größer. Hier kann die Anweisung optimize table Abhilfe schaffen. Dadurch werden alle Datensätze, die als »gelöscht« markiert sind, auch physikalisch gelöscht. Die Datenbank kann somit spürbar verkleinert werden. Für die o. a. Tabelle würde die Anweisung lauten:

```
optimize table personen;
```

drop table

Eine gesamte Tabelle inkl. aller Datensätze kann mit Hilfe von drop table gelöscht werden. Für die o. a. Tabelle würde die Anweisung lauten:

```
drop table personen;
```

Dieser Befehl sollte nur in Ausnahmefällen angewendet werden!

2.4 Tabellenstruktur ändern

alter table

Sollte man feststellen, dass eine Tabelle mit einer nicht ausreichenden oder falschen Struktur erzeugt wurde, so kann diese Struktur mit Hilfe der SQL-Anweisung alter table geändert werden. Je nach Änderung können Verluste bei den bereits eingegebenen Daten auftreten. Darauf wird gesondert hingewiesen. Einige Beispiele:

rename

Name einer Tabelle ändern

```
alter table personen
  rename adr;
```

Der Name der Tabelle personen wird in adr geändert.

change

Name eines Feldes ändern

```
alter table personen
  change vorname vname char(25);
```

Der Name des Feldes vorname in der Tabelle personen wird auf vname geändert. Dieses Feld hat den Datentyp char(25), ohne Angabe des Datentyps kann die Änderung nicht durchgeführt werden.

Länge eines Feldes ändern

```
alter table personen
  change vorname vorname char(35);
```

Die Länge des Feldes vorname in der Tabelle personen wird auf 35 Zeichen geändert (vorher 25). Der Feldname muß doppelt angegeben werden. Bei einer Verkleinerung (z. B. von 25 auf 15 Zeichen) können Daten verloren gehen!

add

Feld hinzufügen

```
alter table personen
  add gehalt double;
```

Das Feld gehalt vom Typ double wird der Tabelle personen hinzugefügt.

Typ eines Feldes ändern

```
alter table personen
  change gehalt gehalt int;
```

Der Typ des Feldes gehalt aus der Tabelle personen wird auf int geändert (vorher double). Der Feldname muß doppelt angegeben werden. Bei einer Veränderung des Typs können Daten verloren gehen (hier sind dies die Nachkommastellen)!

drop

Feld löschen

```
alter table personen
  drop gehalt;
```

Das Feld gehalt der Tabelle personen wird gelöscht. In allen Datensätzen gehen die Daten dieses Feldes verloren!

2.5 Informationen über Datenbanken und Tabellen anzeigen

Falls man nicht weiß, welche Datenbanken und Tabellen bereits existieren bzw. wie eine Tabelle aufgebaut ist, dann kann man auf die SQL-Anweisung show zurückgreifen. Einige Beispiele:

show databases

Alle Datenbanken anzeigen

```
show databases;
```

alle Datenbanken innerhalb von MySQL anzeigen

show tables

Alle Tabellen anzeigen

```
show tables from firma;
```

alle Tabellen der Datenbank firma anzeigen

show columns

Alle Felder anzeigen

```
show columns from personen from firma;
```

alle Felder der Tabelle personen der Datenbank firma anzeigen. Der optionale Zusatz from database (hier from firma) kann weggelassen werden, falls eine Datenbank schon zur Benutzung geöffnet wurde (z. B. mit use).

3 MySQL, Datensätze bearbeiten

3.1 Datensätze erzeugen

insert, values

Nehmen wir an, die Struktur der Tabelle personen entspräche dem Original, das mit create table erzeugt wurde (s. oben). Ein neuer, vollständiger Datensatz für diese Tabelle mit den Feldern name, vorname, personalnummer, gehalt und geburtstag kann dann wie folgt erzeugt werden:

```
insert personen values
(
  'Maier',
  'Hans',
  21398,
  2850.90,
  '1960-12-20'
);
```

Dabei ist Folgendes zu beachten:

- Die Reihenfolge der Feldinhalte in den Klammern hinter values muss der Reihenfolge der erzeugten Felder entsprechen.
- Zeichenketten und Datumsangaben müssen in einfache Hochkommata (Apostroph) gesetzt werden.
- Die Datumsangabe muss im amerikanischen Format erfolgen (JJJJ-MM-TT).

- Bei Zahlen mit Nachkommastellen ist ein Punkt statt eines Kommas zu verwenden (wie bei PHP).

Falls nicht alle Feldinhalte besetzt werden sollen oder die Original-Reihenfolge der Felder nicht beachtet werden soll, kann man auch folgende Form benutzen:

```
insert personen
(
  name,
  gehalt
)
values
(
  'Schmitz',
  3950.90
);
```

Dabei erfolgt eine genaue Zuordnung der neuen Werte zu den genannten Feldern. Die Felder vorname, personalnummer und geburtstag bleiben zunächst leer. Sie sind mit NULL besetzt und können später noch gefüllt werden. NULL bedeutet weder die Zahl 0 noch eine leere Zeichenkette, sondern lediglich »kein Eintrag"!

3.2 Datensätze auswählen und anzeigen

*select, from, **

Die SQL-Anweisung select wird mit Abstand am häufigsten verwendet und dient zur Durchführung einer Datenbank-Abfrage. Es findet eine Auswahl von Datensätzen, die angezeigt werden sollen, statt. Einige Anwendungsbeispiele:

select * from personen; Alle Felder (*) und alle Datensätze der Tabelle personen werden angezeigt. select name, vorname from personen; Die Inhalte der Felder name und vorname aller Datensätze der Tabelle personen werden angezeigt.

where

```
select * from personen
  where gehalt = 3000;
```
Alle Felder und alle Datensätze der Tabelle, bei denen im Feld gehalt der Wert 3000 steht, werden angezeigt.

Hochkommata

```
select * from personen
  where name = 'Maier';
```
Alle Felder und alle Datensätze der Tabelle, bei denen im Feld name 'Maier' steht, werden angezeigt. Zu beachten sind die Hochkommata!

```
select * from personen
  where geburtstag = '1962-1-5';
```
Alle Felder und alle Datensätze der Tabelle, bei denen im Feld geburtstag der 5.1.1962 steht, werden angezeigt. Zu beachten sind die Hochkommata und die Reihenfolge JJJJ-MM-TT!

is null

```
select * from personen
  where vorname is null;
```
Alle Felder und alle Datensätze der Tabelle, bei denen im Feld vorname noch kein Eintrag erfolgte, werden angezeigt.

Übung UE01

Erstellen Sie eine Datenbank für Computer-Hardware. Innerhalb dieser Datenbank soll es eine Tabelle für Festplatten mit folgendem Aufbau und Inhalt geben:

Hersteller	Typ	MB	Preis	Artikel-nummer	Datum der ersten Produktion
Quantum	Fireball CX	6400	215,00	HDA-208	1.10.97
Quantum	Fireball Plus	9100	269,00	HDA-163	15.3.98
Fujitsu	MPE 3136	13600	275,00	HDA-171	1.9.98
Seagate	310232A	10200	245,00	HDA-144	15.11.97
IBM	DJNA 372200	22000	455,00	HDA-140	15.6.98

Gehen Sie dazu wie folgt vor:

- Installieren Sie MySQL (falls noch nicht geschehen).
- Starten Sie den MySQL-Datenbank-Server.
- Rufen Sie den MySQL-Monitor auf.
- Verwenden Sie `create database` zum Erzeugen der Datenbank.
- Wechseln Sie zu dieser Datenbank mit `use`.
- Erzeugen Sie mit `create table` eine Tabelle mit sechs Feldern eines geeigneten Typs.
- Geben Sie mit `insert` den ersten Datensatz ein.
- Kontrollieren Sie mit `select` die korrekte Eingabe der Struktur und des ersten Datensatzes.
- Nehmen Sie gegebenenfalls mit `alter table` Änderungen an der Struktur vor.
- Geben Sie die restlichen Datensätze mit `insert` ein.
- Kontrollieren Sie mit `select` die korrekte Eingabe der gesamten Tabelle.
- Beenden Sie den MySQL-Monitor.
- Fahren Sie den MySQL-Datenbank-Server wieder herunter.

Sollten Sie Fehler bei der Eingabe der Datensätze gemacht haben, so können Sie diese später noch verändern oder löschen.

3.3 Vergleichs-Operatoren, logische Operatoren

Vergleichs-Operatoren

Bei der Auswahl durch where innerhalb der select-Anweisung (und später auch in anderen Anweisungen) können, ähnlich wie bei der Programmierung mit PHP, die nachfolgenden Vergleichs-Operatoren angewendet werden:

Operator	Bedeutung
=	gleich
<>	ungleich
>	größer als
>=	größer als oder gleich
<	kleiner als
<=	kleiner als oder gleich

Logische Operatoren

Es können auch mehrere Auswahlbedingungen logisch miteinander verknüpft werden, und zwar mit Hilfe der folgenden logischen Operatoren:

Operator	Bedeutung
not	Der Wahrheitswert einer Bedingung wird umgekehrt.
and	Alle Bedingungen müssen zutreffen.
or	Eine der Bedingungen muss zutreffen.

Es folgt ein Beispiel für die Anwendung von Vergleichsoperatoren und logischen Operatoren. Die nachfolgende Auswahl ergibt alle Datensätze, bei denen im Feld gehalt ein Wert zwischen 3000 und 3500 steht:

```
select * from personen
  where gehalt >= 3000 and gehalt <= 3500;
```

3.4 Vergleichsoperator like

like, %, _

Der Operator like ist sehr nützlich beim Suchen nach Zeichenketten oder Teilen von Zeichenketten. Dabei können auch Platzhalter (Wildcards) eingesetzt werden. Ein % (Prozentzeichen) steht für eine beliebige Anzahl von unbekannten Zeichen, ein _ (Unterstrich) steht für genau ein unbekanntes Zeichen. Die untersuchte Zeichenkette muss dabei nach wie vor in einfache Hochkommata gesetzt werden. Einige Beispiele:

```
select * from fp
  where typ like 'a%';
```

Alle Felder und alle Datensätze der Tabelle werden angezeigt, deren Eintrag im Feld typ mit »a« oder »A« beginnt.

```
select * from fp
  where typ like '%a';
```

Alle Felder und alle Datensätze der Tabelle werden angezeigt, deren

Eintrag im Feld typ mit »a« oder »A« endet. select * from fp
where typ like '%a%'; Alle Felder und alle Datensätze der Tabelle werden angezeigt, deren Eintrag im Feld typ ein »a« oder »A« beinhaltet. select * from fp
where typ like 'D_NA%'; Alle Felder und alle Datensätze der Tabelle werden angezeigt, deren Eintrag im Feld typ mit »D(beliebiges Zeichen)NA« beginnt. Dies gilt unabhängig von Groß- und Kleinschreibung. Angezeigt werden also z. B. DJNA..., DTNA..., aber nicht DTTA...

3.5 Sortierung

order by, desc

Zusätzlich läßt sich die Reihenfolge der Ausgabe mit Hilfe von order by beeinflussen. Die folgende SQL-Anweisung sortiert die Ausgabe nach Gehalt, beginnend mit dem größten Gehalt:

```
select * from personen
  order by gehalt desc;
```

Der Zusatz desc steht für descending (=absteigend). Im Normalfall wird aufsteigend sortiert, der Zusatz asc für ascending (=aufsteigend) muss deshalb nicht gesondert erwähnt werden.

Übung UE02

Führen Sie zu der Hardware-Datenbank, die mit der vorherigen Übung erzeugt wurde, verschiedene Abfragen durch. Es sollen alle Festplatten angezeigt werden, die folgende Kriterien erfüllen:

- des Herstellers Quantum, mit allen Angaben
- mit einer Kapazität von mehr als 10 000 MB, nur mit den Angaben Hersteller, Typ, MB
- mit einem Preis von weniger als 300 DM, nur mit den Angaben Hersteller, Preis, Artikelnummer, nach Preis aufsteigend sortiert
- mit einer Kapazität von mehr als 10 000 MB, die weniger als 300 DM kosten, mit allen Angaben, nach MB absteigend sortiert
- deren Typbezeichnung mit »Fire« beginnt, mit allen Angaben
- in deren Typbezeichnung »CX« vorkommt, mit allen Angaben
- die nach dem 1.1.98 erstmalig produziert wurden, mit allen Angaben
- die im ersten Halbjahr 1998 erstmalig produziert wurden, mit allen Angaben

3.6 Datensätze ändern

update, set

Die SQL-Anweisung update dient zur Änderung von einem oder mehreren Feldinhalten in einem oder mehreren vorhandenen Datensätzen. Sie ähnelt vom Aufbau her der select-Anweisung. Man sollte unbedingt darauf achten, dass die Auswahlkriterien sorgfältig

gewählt werden, da ansonsten evtl. nicht nur die gewünschten Datensätze verändert werden. Einige Beispiele:

Absolute Änderung

```
update fp
  set preis = 300
  where hersteller = 'IBM';
```

Bei allen Datensätzen, deren Eintrag im Feld Hersteller ’IBM’ lautet, wird der Preis auf den Wert 300 gesetzt.

Relative Änderung (mit Berechnungsformel)

```
update fp
  set preis = preis * 1.1
  where hersteller = 'IBM';
```

Bei allen Datensätzen, deren Eintrag im Feld Hersteller ’IBM’ lautet, wird der Preis um 10% erhöht.

Mehrere Änderungen gleichzeitig

```
update fp
   set preis = 300, hersteller = 'IBM Corp.'
   where hersteller = 'IBM';
```

Bei allen Datensätzen, deren Eintrag im Feld Hersteller 'IBM' lautet, wird der Preis auf 300 und der Eintrag im Feld Hersteller auf 'IBM Corp.' gesetzt.

3.7 Datensätze löschen

delete

Die SQL-Anweisung delete dient zum Löschen von einem oder mehreren vorhandenen Datensätzen. Sie ähnelt ebenfalls vom Aufbau her der select-Anweisung und sollte sehr umsichtig eingesetzt werden, da ansonsten evtl. nicht nur die gewünschten Datensätze gelöscht werden. Nachdem z. B. zu einem früheren Zeitpunkt mit der folgenden Anweisung ein Datensatz eingefügt wurde:

```
insert fp
(
  hersteller,
  typ,
  artnummer
)
values
(
  'Western Digital',
  'WD-102AA',
  'HDA-178'
);
```

könnte dieser Datensatz wieder gelöscht mit:

```
delete from fp
  where artnummer = 'HDA-178';
```

Es empfiehlt sich, dabei ein möglichst eindeutiges Kriterium zu verwenden, da ansonsten nicht nur die gewünschten Datensätze gelöscht werden. Im vorliegenden Falle ist dies die Artikelnummer.

Falls man unvollständige oder leere Datensätze einer Tabelle hinzugefügt hat, so kann man sie z. B. mit folgender Anweisung löschen:

```
delete from fp
  where hersteller is null;
```

Diese Anweisung löscht alle Datensätze, die keinen Eintrag im Feld hersteller haben.

Übung UE03

Führen Sie in der Hardware-Datenbank aus den vorherigen Übungen folgende Änderungen bzw. Löschungen durch:

- Alle Festplatten des Herstellers Seagate sollen um 25 DM teurer werden.
- Der Hersteller Fujitsu hat seine Festplatte MPE 3136 ersetzt durch die Festplatte MPE 3139 mit einer Kapazität von 13900 MB. Führen Sie die notwendige Änderung durch.
- Alle Festplatten des Herstellers Quantum sollen um 8 % teurer werden.
- Der Hersteller IBM hat seine Produktion eingestellt. Entfernen Sie die betreffenden Festplatten aus Ihrem Angebot.
- Löschen Sie alle versehentlich eingetragenen Datensätze. Das Kriterium für einen solchen Datensatz soll sein: Der Datensatz hat keine Artikelnummer.

Zur Kontrolle: Nach den Änderungen auf der Basis der Originaldaten (siehe erste Übung) sollte die Datenbank folgendes Aussehen haben:

Hersteller	Typ	MB	Preis	Artikel-nummer	Datum der ersten Produktion
Quantum	Fireball CX	6400	232,20	HDA-208	1.10.97
Quantum	Fireball Plus	9100	290,52	HDA-163	15.3.98
Fujitsu	MPE 3139	13900	275,00	HDA-171	1.9.98
Seagate	310232A	10200	270,00	HDA-144	15.11.97

4 Eindeutigkeit von Datensätzen, Index

Indizierung

Für viele Vorgänge innerhalb von Tabellen ist eine eindeutige Identifizierung der einzelnen Datensätze hilfreich und notwendig. Dies wird mit Hilfe eines sogenannten »eindeutigen Index« realisiert. Dieser Index kann unmittelbar beim Erzeugen einer neuen Tabelle geschaffen werden (mit create table).

Er kann aber auch einer vorhandenen Tabelle, die bereits Daten beinhaltet, hinzugefügt werden (mit alter table). Es empfiehlt sich allerdings, sich bereits vor Erzeugung der Tabelle zu überlegen, welches Feld eindeutige Daten beinhaltet, und die Tabelle mit eindeutigem Index zu erzeugen.

4.1 Neue Tabelle mit Index erzeugen

index, unique

Die Erzeugung einer Tabelle mit einem eindeutigen Index wird an folgendem Beispiel gezeigt:

```
create table p
(
  nachname char(30),
  nummer int not null,
  index (nummer),
  unique (nummer)
);
```

Diese SQL-Anweisung erzeugt eine Tabelle mit dem Namen p und den zwei Feldern nachname und nummer. Das Feld nummer ist vom Typ int (für ganze Zahlen). Neue Datensätze müssen für dieses Feld einen Eintrag erhalten, dies ist mit dem Zusatz not null festgelegt worden. Dieser Zusatz ist für den Index notwendig.

Mit index (nummer) wird ein Index auf dem Feld nummer erzeugt. Der Index bekommt auch den Namen nummer und wird mit unique (nummer) zu einem eindeutigen Index.

Anschließend können in dieser Tabelle nur noch Einträge aufgenommen werden, die einen Eintrag im Feld nummer haben. Dieser Eintrag darf nicht bereits in einem anderen Datensatz existieren, da ansonsten die Eindeutigkeit verletzt wäre. Alle Datensätze können anhand des Wertes im Feld nummer eindeutig voneinander unterschieden werden. Dies ist besonders beim Ändern und Löschen von Datensätzen wichtig.

Primary Key

Mit Hilfe der Anweisung show columns from p; kann man sich die Struktur der Tabelle anschauen. Beim Feld nummer ist in der Spalte Key der Eintrag PRI zu sehen. Dies bedeutet, daß für diese Spalte der **Primary Key** erzeugt wurde. Der **Primary Key** ist ein eindeutiger Index.

4.2 Index zu vorhandener Tabelle hinzufügen

Einer vorhandenen Tabelle kann ein Index hinzugefügt werden. Dazu sind mehrere Schritte notwendig. Nehmen wir an, die Tabelle wäre mit der folgenden SQL-Anweisung erzeugt worden:

```
create table p
(
  nachname char(30),
  nummer int
);
```

Zunächst muß das Feld nummer nachträglich auf die Eigenschaft not null geändert werden. Dabei sollte vorher darauf geachtet werden, daß in keinem der vorhandenen Datensätze ein leerer Eintrag im Feld nummer existiert, ansonsten treten Fehler bei der Veränderung der Feldeigenschaften auf. Die SQL-Anweisung für diese Änderung lautet:

alter ... change not null

```
alter table p
  change nummer nummer int not null;
```

Anschließend kann ein eindeutiger Index für das Feld nummer erzeugt werden. Darüber hinaus sollte darauf geachtet werden, daß im Feld nummer kein Eintrag mehr als einmal existiert, ansonsten wird der eindeutige Index nicht erzeugt. Die SQL-Anweisung für die Änderung lautet:

alter ... add unique

```
alter table p
  add unique (nummer);
```

Man sollte sich mit Hilfe von show columns from p; davon überzeugen, daß der **Primary Key** für das Feld nummer existiert.

4.3 Index löschen

Sollte man versehentlich das falsche Feld für einen eindeutigen Index gewählt haben, so lässt er sich mit der folgenden SQL-Anweisung wieder entfernen (für o. a. Beispiel):

```
alter table p
  drop primary key;
```

Falls man darüber hinaus für dieses Feld auch leere Einträge erlauben möchte, muss die betreffende Feld-Eigenschaft noch geändert werden:

```
alter table p
  change nummer nummer int null;
```

Übung UE04

Erzeugen Sie für die Tabelle personen in der Datenbank firma einen eindeutigen Index auf dem Feld personalnummer. Lassen Sie sich vorher alle Datensätze anzeigen, und nehmen Sie gegebenenfalls notwendige Änderungen bezüglich der Inhalte des Feldes personalnummer vor.

Überzeugen Sie sich anschließend davon, dass der eindeutige Index erzeugt wurde: Versuchen Sie einen Datensatz einzutragen, der einen bereits vorhandenen Eintrag im Feld personalnummer hat.

Übung UE05

Erzeugen Sie für die Tabelle fp in der Datenbank hardware einen eindeutigen Index auf dem Feld artnummer. Lassen Sie sich vorher alle Datensätze anzeigen, und nehmen Sie gegebenenfalls notwendige Änderungen bezüglich der Inhalte des Feldes artnummer vor.

Überzeugen Sie sich anschließend davon, dass der eindeutige Index erzeugt wurde: Versuchen Sie einen Datensatz einzutragen, der einen bereits vorhandenen Eintrag im Feld artnummer hat.

5 Hilfsmittel für MySQL-Datenbanken

PHPMyAdmin

Ein nützliches Hilfsmittel zur übersichtlichen Administration von umfangreicheren Datenbanken ist das Programm **PHPMyAdmin** von Tobias Ratschiller. Es steht unter der Adresse http://www.phpwizard.net/phpMyAdmin als Freeware-Tool zur Verfügung.

E.6 Alle MySQL-Befehle

Alle MySQL-Befehle

Im vorliegenden Abschnitt werden eine ganze Reihe von Befehlen für MySQL eingesetzt. Zur besseren Übersicht schließt sich eine Liste aller Befehle mit einer kurzen Erläuterung derselben an. Weitergehende Ausführungen finden sich im MySQL-Manual.

Funktionsname	Erläuterung
alter table	Ändern der Struktur einer Tabelle
create database	Erzeugen einer Datenbank
create function	Erzeugen einer benutzerdefinierten Funktion
create index	Erzeugen eines Index in einer Tabelle
create table	Erzeugen einer Tabelle
delete	Löschen von Datensätzen
describe	zeigt das Ergebnis, das eine Auswahl mit **select** erzeugen würde (s. **explain**)
drop database	Löschen einer Datenbank
drop function	Löschen einer benutzerdefinierten Funktion
drop index	Löschen eines Index in einer Tabelle
drop table	Löschen einer Tabelle
explain	zeigt das Ergebnis, das eine Auswahl mit **select** erzeugen würde (s. **describe**)
flush	Löschen von MySQL-internen Caches
grant	Vergabe von Benutzerrechten innerhalb einer Datenbank
insert	Erzeugen von neuen Datensätzen
join	dient zur Auswahl von Datensätzen aus mehreren Tabellen mit **select**
kill	Löschen eines Threads, in dem eine MySQL-Verbindung aktiv ist
load data	Erzeugen von neuen Datensätzen durch Einlesen aus einer Textdatei
lock tables	Sperren von Tabellen im aktiven Thread
optimize table	Optimieren einer Tabelle, Freigabe von nicht mehr genutztem Speicherplatz
replace	Ersetzen von Datensätzen durch neue Datensätze

revoke	Rücknahme von Benutzerrechten innerhalb einer Datenbank
select	Auswahl von Datensätzen
set option	Setzen von Optionen der aktuellen Datenbank-Verbindung
show columns	Anzeige der Felder einer Tabelle
show databases	Anzeige der Namen aller Datenbanken von MySQL
show grants	Anzeige aller Rechte eines Benutzers der Datenbank
show index	Anzeige der Indizes einer Tabelle
show processlist	Anzeige aller Threads, in denen eine MyQSL-Verbindung aktiv ist
show status	Anzeige von Status-Informationen des MySQL-Datenbankservers
show table status	Anzeige von Informationen zu den Tabellen einer Datenbank
show tables	Anzeige der Namen aller Tabellen einer Datenbank
show variables	Anzeige des Wertes der MySQL-System-Variablen
unlock tables	Rücknahme der Sperrung von Tabellen im aktiven Thread
update	Ändern von Datensätzen
use	Benutzen einer Datenbank, Wechseln zu einer Datenbank