

Einführung in MySQL

Bernd Wurst

schokoeks.org Webhosting
bernd@schokoeks.org

Vortrag Linux-User-Group Backnang
11. Januar 2007

Inhalt

- I. Begriffe und Konventionen
 - II. Einfache Tabellen und Abfragen
 - III. Relationen und Fremdschlüssel
 - IV. Subselects
 - V. JOINS
 - VI. VIEWs
 - VII. Transaktionen
-
-

Einschlägige Begriffe

- SQL → Structured Query Language
(auch: Simple Query Language)
 - RDBMS → relational database
management system
 - Transaktionen
 - Relationen
 - NULL → Nix
-
-

Konventionen

- SQL ist (englischer) Fließtext
 - Sprachbestandteile werden groß geschrieben
 - Namen werden klein geschrieben
 - Ob ein oder mehrere Leerzeichen oder Zeilenumbrüche benutzt werden ist egal
 - Wir verwenden MySQL 5
-
-

SQL oder MySQL?

- Anführungszeichen
 - MySQL: ``table`='foobar'` oder ``table`="foobar"`
 - Standard-SQL: `"table"='foobar'`
 - Datentypen
 - MySQL kann kein boolean
 - SQLite kann kein Datum
 - ⇒ portable Anwendungen können die Funktionen der einzelnen Datenbanken nicht optimal nutzen
 - Manche Bindings haben Probleme mit Zeichensätzen
-
-

Einfache Tabellen (Schema)

Linux-Benutzeraccounts:

Field	Type	Null	Key	Default	Extra
uid	int(10) unsigned	NO	PRI	NULL	auto_increment
name	varchar(255)	YES		NULL	
username	varchar(50)	NO	UNI		
gid	int(11)	NO		100	
homedir	varchar(255)	NO			
shell	varchar(25)	NO		/bin/bash	

Einfache Tabellen (Schema)

Linux-Benutzeraccounts:

Field	Type	Null	Key	Default	Extra
uid	int(10) unsigned	NO	PRI	NULL	auto_increment
username	varchar(50)	NO	UNI		
gid	int(11)	NO		100	
homedir	varchar(255)	NO			
shell	varchar(25)	NO		/bin/bash	

Bedeutung:

„Feld »uid« (für »user-ID«) ist (ohne Ausnahme) eine Zahl (10 Bit) ohne Vorzeichen. Dieses Feld ist der Primärschlüssel und wird bei neuen Einträgen automatisch auf einen eindeutigen Wert gesetzt.“

Einfache Tabellen (Schema)

Linux-Benutzeraccounts:

Field	Type	Null	Key	Default	Extra
uid	int(10) unsigned	NO	PK1	NULL	auto_increment
name	varchar(255)	YES		NULL	
username	varchar(50)	NO	UNI		
gid	int(11)	NO		100	
homedir	varchar(255)	NO			
shell	varchar(25)	NO		/bin/bash	

Bedeutung:

„Feld »name« ist ein Text mit maximal 255 Zeichen.
Dieses Feld kann auch leer bleiben.“

Einfache Tabellen (Schema)

Linux-Benutzeraccounts:

Field	Type	Null	Key	Default	Extra
uid	int(10) unsigned	NO	PRI	NULL	auto increment
name	varchar(255)	YES		NULL	
username	varchar(50)	NO	UNI		
gid	int(11)	NO		100	
homedir	varchar(255)	NO			
shell	varchar(25)	NO		/bin/bash	

Bedeutung:

„Feld »username« ist ein Text mit maximal 50 Zeichen. Dieses Feld kann nicht leer bleiben. Ein Index über dieses Feld ist angelegt und es kann jeder Wert nur einmal vorkommen“

Einfache Tabellen (Schema)

Linux-Benutzeraccounts:

Field	Type	Null	Key	Default	Extra
uid	int(10) unsigned	NO	PRI	NULL	auto_increment
name	varchar(255)	YES		NULL	
username	varchar(50)	NO	UNI		
gid	int(11)	NO		100	
homedir	varchar(255)	NO			
shell	varchar(25)	NO		/bin/bash	

Bedeutung:

„Feld »gid« (für »Group-ID«) ist (ohne Ausnahme) eine Zahl (11 Bit mit Vorzeichen). Wird nichts angegeben, erhalten neue Einträge dort den Wert »100«.“

Einfache Tabellen (Schema)

Field	Type	Null	Key	Default	Extra
uid	int(10) unsigned	NO	PRI	NULL	auto_increment
name	varchar(255)	YES		NULL	
username	varchar(50)	NO	UNI		
gid	int(11)	NO		100	
homedir	varchar(255)	NO			
shell	varchar(25)	NO		/bin/bash	

Erstellen obiger Tabelle:

```
CREATE TABLE useraccounts (  
  uid INT(10) UNSIGNED NOT NULL AUTO_INCREMENT,  
  name VARCHAR(255) NULL,  
  username VARCHAR(50) NOT NULL,  
  gid INT(11) NOT NULL DEFAULT 100,  
  homedir VARCHAR(255) NOT NULL,  
  shell VARCHAR(25) NOT NULL DEFAULT '/bin/bash',  
  
  PRIMARY KEY (uid),  
  UNIQUE (username)  
) TYPE=InnoDB AUTO_INCREMENT=1001;
```

INSERT

Daten in Tabelle einfügen:

```
mysql> INSERT INTO useraccounts  
  (username, homedir)  
VALUES  
  ('bernd', '/home/bernd');
```

```
mysql> INSERT INTO useraccounts  
  (username, homedir)  
VALUES  
  ('bernd', '/home/bernd');
```

```
ERROR 1062 (23000): Doppelter Eintrag 'bernd' fuer Schluessel 2
```

SELECT

Einfache SQL-Abfrage-Befehle sind Dreigliedrig:
Was?
Woher?
Welche? (kann weggelassen werden)

Syntax:

SELECT

[was?]

← *Feld-Namen oder Daten*

FROM

[woher?]

← *Tabelle(n)*

WHERE

[welche?]

← *Einschränkung*

Einfache Tabellen (Daten)

Alle Daten auslesen:

```
mysql> SELECT * FROM useraccounts;
```

uid	name	username	gid	homedir	shell
1001	NULL	bernd	100	/home/bernd	/bin/bash
1008	NULL	hanno	100	/home/hanno	/bin/bash
1094	Linux User Group Backnang	lug-bk	100	/home/lug-bk	/bin/bash
1301	NULL	otih	100	/home/otih	/bin/bash

Bedeutung:

„Hole alle Felder aus der Tabelle »useraccounts«.“

Einfache Tabellen (Daten)

Gleichbedeutend mit:

```
mysql> SELECT uid, name, username, gid,  
        homedir, shell FROM useraccounts;
```

uid	name	username	gid	homedir	shell
1001	NULL	bernd	100	/home/bernd	/bin/bash
1008	NULL	hanno	100	/home/hanno	/bin/bash
1094	Linux User Group Backnang	lug-bk	100	/home/lug-bk	/bin/bash
1301	NULL	otih	100	/home/otih	/bin/bash

Bedeutung:

„Hole die Felder »uid«, »name«, »username«, »gid«, »homedir« und »shell« aus der Tabelle »useraccounts«.“

Einfache Tabellen (Daten)

Zeige nur User-IDs und Benutzernamen

```
mysql> SELECT uid, username FROM  
useraccounts;
```

```
+-----+-----+  
| uid   | username |  
+-----+-----+  
| 1001  | bernd   |  
| 1008  | hannu   |  
| 1094  | lug-bk  |  
| 1301  | oti     |  
+-----+-----+
```

Einfache Tabellen (Abfragen)

```
mysql> SELECT * FROM useraccounts LIMIT 2;
```

uid	name	username	gid	homedir	shell
1001	NULL	bernd	100	/home/bernd	/bin/bash
1008	NULL	hanno	100	/home/hanno	/bin/bash

Bedeutung:

„Hole alle Felder aus der Tabelle »useraccounts«, hör aber auf nach 2 Zeilen der Ausgabe.“

Einfache Tabellen (Abfragen)

```
mysql> SELECT * FROM useraccounts WHERE  
      name IS NULL;
```

uid	name	username	gid	homedir	shell
1001	NULL	bernd	100	/home/bernd	/bin/bash
1008	NULL	hanno	100	/home/hanno	/bin/bash
1301	NULL	otih	100	/home/otih	/bin/bash

Bedeutung:

„Hole alle Felder aus der Tabelle »useraccounts«, deren Name nicht gesetzt (also NULL) ist.“

Einfache Tabellen (Abfragen)

Es ist egal, ob die Spalten der Beschränkung auch angezeigt werden:

```
mysql> SELECT uid, username FROM  
        useraccounts WHERE name IS NULL;
```

```
+-----+-----+  
| uid   | username |  
+-----+-----+  
| 1001  | bernd   |  
| 1008  | hannu   |  
| 1301  | otih    |  
+-----+-----+
```

SELECT

Nicht auf Datenbank-Inhalte begrenzt:

```
mysql> SELECT 1+1;
```

```
+-----+  
| 1+1 |  
+-----+  
| 2 |  
+-----+
```

```
mysql> SELECT DATE(NOW()) + INTERVAL 2  
MONTH;
```

```
+-----+  
| DATE(NOW()) + INTERVAL 2 MONTH |  
+-----+  
| 2008-03-11 |  
+-----+
```

SELECT (Aggregationen)

```
mysql> SELECT SUM(uid)
        FROM useraccounts;
```

```
+-----+
| SUM(uid) |
+-----+
|      4404 |
+-----+
```

```
mysql> SELECT MIN(uid)
        FROM useraccounts;
```

```
+-----+
| MIN(uid) |
+-----+
|      1001 |
+-----+
```

```
mysql> SELECT AVG(uid)
        FROM useraccounts;
```

```
+-----+
| AVG(uid) |
+-----+
|      1101 |
+-----+
```

```
mysql> SELECT MAX(uid)
        FROM useraccounts;
```

```
+-----+
| MAX(uid) |
+-----+
|      1301 |
+-----+
```

Zusammenfassung bisher

Was wir bisher können:

- Tabelle erstellen
- Daten einfügen
- Daten wieder auslesen
 - Wahlweise auch nur bestimmte Felder oder Datensätze oder aggregierte Werte

Wir wollen mehr:

- z.B. Daten verknüpfen
 - Oder komplexe Abfragen speichern
-
-

Relationen (einfach)

useraccounts:

Field	Type	Null	Key	Default	Extra
uid	int(10) unsigned	NO	PRI	NULL	auto_increment
name	varchar(255)	YES		NULL	
username	varchar(50)	NO	MUL		
gid	int(11)	NO		100	
homedir	varchar(255)	NO			
shell	varchar(25)	NO		/bin/bash	

groups:

Field	Type	Null	Key	Default	Extra
gid	int(11)	NO	PRI		
name	varchar(50)	NO	UNI		
notizen	text	YES			

```
mysql> ALTER TABLE useraccounts ADD FOREIGN KEY (gid)
REFERENCES groups(gid);
```

Subselects

```
SELECT
  u.username AS username,
  (SELECT name FROM groups
   WHERE groups.gid = u.gid
  ) AS hauptgruppe
FROM useraccounts AS u;
```

username	hauptgruppe
bernd	users
hanno	users
lug-bk	users
otih	users

JOINS

```
SELECT
    u.username AS username,
    g.name AS hauptgruppe
FROM useraccounts AS u
    LEFT JOIN groups AS g
        USING (gid);
```

username	hauptgruppe
bernd	users
hanno	users
lug-bk	users
otih	users

Relationen (many-to-many)

Field	Type	Null	Key	Default	Extra
uid	int(10) unsigned	NO	PRI	NULL	auto_increment
name	varchar(255)	YES		NULL	
username	varchar(50)	NO	MUL		
gid	int(11)	NO		100	
homedir	varchar(255)	NO			
shell	varchar(25)	NO		/bin/bash	

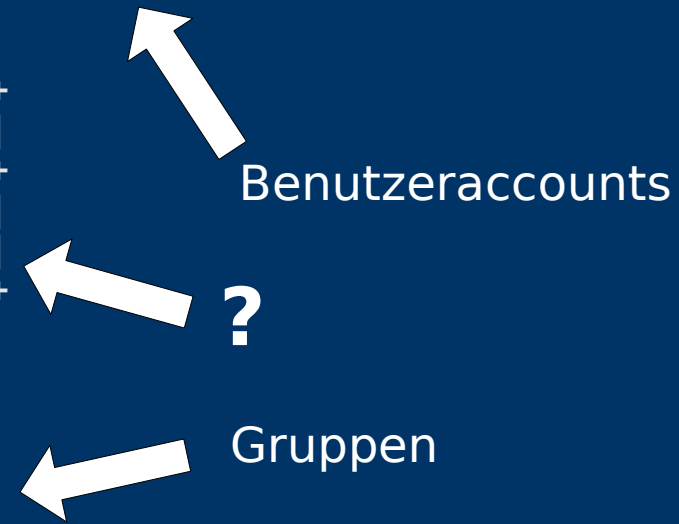
Field	Type	Null	Key	Default	Extra
gid	int(11)	NO	MUL		
uid	int(10) unsigned	NO	MUL		

Field	Type	Null	Key	Default	Extra
gid	int(11)	NO	PRI		
name	varchar(50)	NO	UNI		
notizen	text	YES			

Benutzeraccounts

?

Gruppen



JOINS

```
SELECT
  u.username AS username,
  g.name AS gruppe
FROM useraccounts AS u
LEFT JOIN user_groups AS ug
  USING (uid)
LEFT JOIN groups AS g
  ON (g.gid=ug.gid)
WHERE u.username='lug-bk';
```

Field	Type	Null	Key	Default
uid	int(10) unsigned	NO	PRI	NULL
name	varchar(255)	YES		NULL
username	varchar(50)	NO	MUL	
gid	int(11)	NO		100
homedir	varchar(255)	NO		
shell	varchar(25)	NO		/bin/ba

Field	Type	Null	Key	Default	Extra
gid	int(11)	NO	MUL		
uid	int(10) unsigned	NO	MUL		

Field	Type	Null	Key	Default	Extra
gid	int(11)	NO	PRI		
name	varchar(50)	NO	UNI		
notizen	text	YES			

username	gruppe
lug-bk	mailusers
lug-bk	cron
lug-bk	users
lug-bk	ssh
lug-bk	backup

VIEWS

„Speichern“ einer Abfrage:

```
CREATE VIEW v_user_groups AS  
SELECT  
  u.username AS username,  
  g.name AS gruppe  
FROM useraccounts AS u  
  LEFT JOIN user_groups AS ug  
    USING (uid)  
  LEFT JOIN groups AS g  
    ON (g.gid=ug.gid)
```

VIEWs

VIEW kann wie eine normale Tabelle abgefragt werden:

```
mysql> SELECT * FROM v_user_groups  
WHERE username='lug-bk';
```

```
+-----+-----+  
| username | gruppe |  
+-----+-----+  
| lug-bk   | mailusers |  
| lug-bk   | cron      |  
| lug-bk   | users     |  
| lug-bk   | ssh       |  
| lug-bk   | backup    |  
+-----+-----+
```

Transaktionen

- Transaktionen sind SQL-Befehle, die zusammengefasst werden (siehe ACID)
 - Nicht alle Befehle können in Transaktionen angewendet werden
 - Strukturbefehle
 - Administrative Befehle
 - Auch Lese-Operationen können von Transaktionen beeinflusst werden
 - MySQL macht per Default keine Transaktionen
-
-

Transaktions-Eigenschaften (ACID)

- **atomicity - Atomarität**
Eine Transaktion wird ganz oder gar nicht ausgeführt
 - **consistency - Konsistenz**
Eine Transaktion führt die Datenbank von einem konsistenten in einen (anderen) konsistenten Zustand
 - **isolation - Isoliertheit**
Eine Transaktion beeinflusst eine weitere, parallel laufende Transaktion nicht
 - **durability - Dauerhaftigkeit**
Das Ergebnis einer Transaktion ist dauerhaft
-
-

Transaktionen

```
mysql> START TRANSACTION;
mysql> INSERT INTO useraccounts (username,
  homedir) VALUES ('test', '/home/test');
mysql> SELECT * FROM useraccounts WHERE
  username='test';
```

```
+-----+-----+-----+-----+-----+-----+
| uid  | name | username | gid | homedir   | shell   |
+-----+-----+-----+-----+-----+-----+
| 1002 | NULL | test     | 100 | /home/test | /bin/bash |
+-----+-----+-----+-----+-----+-----+
```

```
mysql> ROLLBACK;
mysql> SELECT * FROM useraccounts WHERE
  username='test';
Empty set (0.00 sec)
```

Transaktionen

```
mysql> START TRANSACTION;  
mysql> INSERT INTO useraccounts (username,  
  homedir) VALUES ('test', '/home/test');  
mysql> SELECT * FROM useraccounts WHERE  
  username='test';
```

```
+-----+-----+-----+-----+-----+-----+  
| uid  | name | username | gid | homedir  | shell  |  
+-----+-----+-----+-----+-----+-----+  
| 1002 | NULL | test     | 100 | /home/test | /bin/bash |  
+-----+-----+-----+-----+-----+-----+
```

```
mysql> COMMIT;  
mysql> SELECT * FROM useraccounts WHERE  
  username='test';
```

```
+-----+-----+-----+-----+-----+-----+  
| uid  | name | username | gid | homedir  | shell  |  
+-----+-----+-----+-----+-----+-----+  
| 1002 | NULL | test     | 100 | /home/test | /bin/bash |  
+-----+-----+-----+-----+-----+-----+
```

Transaktionen

```
mysql> START TRANSACTION;  
mysql> SELECT * FROM useraccounts WHERE  
    username='test';
```

```
+-----+-----+-----+-----+-----+-----+  
| uid  | name | username | gid | homedir   | shell   |  
+-----+-----+-----+-----+-----+-----+  
| 1002 | NULL | test     | 100 | /home/test | /bin/bash |  
+-----+-----+-----+-----+-----+-----+
```

Andere
Datenbankverbindung



```
mysql> DELETE FROM useraccounts;  
mysql> SELECT * FROM useraccounts  
    WHERE username='test';  
Empty set (0.00 sec)
```

```
mysql> SELECT * FROM useraccounts WHERE  
    username='test';
```

```
+-----+-----+-----+-----+-----+-----+  
| uid  | name | username | gid | homedir   | shell   |  
+-----+-----+-----+-----+-----+-----+  
| 1002 | NULL | test     | 100 | /home/test | /bin/bash |  
+-----+-----+-----+-----+-----+-----+
```

Transaktionen

Stolpersteine:

- Manche Bindings (z.B. Python) starten automatisch eine Transaktion bei Verbindung zu InnoDB-Tabellen
 - Wenn MyISAM-Tabellen beteiligt sind, werden die ACID-Eigenschaften nicht eingehalten
 - Wenn ein Struktur-Befehl erteilt wird, wird die Transaktion vorher abgeschlossen.
-
-

phpMyAdmin

Für Änderungen an der Datenbank-
Struktur eignet sich das Web-Tool
phpMyAdmin sehr gut:
Befehle, die man selten braucht

phpMyAdmin benutzt auch normale SQL-
Befehle und zeigt diese an:
Ergibt Lerneffekt

Aber: Umständlicher bei komplexen
Abfragen

Vielen Dank für Ihre Aufmerksamkeit

Die Vortragsfolien gibt es nach der Veranstaltung online unter
www.lug-bk.de

