

Der Vorteil dieser Methode der Rechtfestlegung mit SQL-Befehlen gegenüber der direkten Bearbeitung der Rechtedatenbank mysql ist, dass der Administrator die Inhalte der Tabellen dieser Rechtedatenbank nicht kennen muss und nicht entscheiden muss, in welche dieser Tabellen die Einträge verteilt werden müssen.

Die Rechte der einzelnen Benutzer können in der Benutzerübersicht des Clients phpmyadmin eingesehen werden.

Benutzer	Host	Passwort	Globale Rechte	GRANT	Aktion
Jeder	%	-	USAGE	Nein	Rechte ändern Exportieren
Jeder	localhost	Nein	USAGE	Nein	Rechte ändern Exportieren
Angestellter	212.211.130.%	Ja	SELECT, INSERT, UPDATE	Nein	Rechte ändern Exportieren
pma	localhost	Nein	USAGE	Nein	Rechte ändern Exportieren
root	127.0.0.1	Nein	ALL PRIVILEGES	Ja	Rechte ändern Exportieren
root	localhost	Nein	ALL PRIVILEGES	Ja	Rechte ändern Exportieren
schueler	%rbs- ulm.de	Ja	USAGE	Nein	Rechte ändern Exportieren

Die Rechte des Benutzers `schueler` des Beispiels sind mit dem Begriff `USAGE` beschrieben. Dieses Schlüsselwort sagt aus, dass der Benutzer `schueler` keine globalen Rechte innerhalb dieses Datenbankservers besitzt.

Seine Rechte auf die Datenbank `faradiso` sind in der Tabelle `db` der Datenbank `mysql` hinterlegt. Hier ein Ausschnitt der Einträge in dieser Tabelle:

Host	Db	User	Select_priv	Insert_priv	Update_priv	Delete_priv	Create_priv	Drop_priv	Grant_priv
%	test		Y	Y	Y	Y	Y	Y	N
%	test_%		Y	Y	Y	Y	Y	Y	N
localhost	phpmyadmin	pma	Y	Y	Y	Y	N	N	N
%rbs- ulm.de	faradiso	schueler	Y	Y	Y	Y	Y	Y	N

Der Schüler hat z. B. nicht das Recht `Grant_priv` erhalten (Eintrag ist N), da er sonst anderen Benutzern Rechte zuweisen könnte und somit seine Beschränkungen umgehen könnte.

Damit der Benutzer `schueler` z. B. keine Tabellen oder Datenbanken löschen kann, wird ihm das Recht `Drop_priv` entzogen. Dies geschieht durch Bearbeiten seiner Rechte im Client `phpmyadmin`. Dort wird der Button `N` in der Zeile `Drop_priv` ausgewählt und bestätigt.

Spalte	Typ	Funktion	Null	Wert
Host	char(60)			%rbs- ulm.de
Db	char(64)			faradiso
User	char(16)			schueler
Select_priv	enum	--	<input type="radio"/> N <input checked="" type="radio"/> Y	
Insert_priv	enum	--	<input type="radio"/> N <input checked="" type="radio"/> Y	
Update_priv	enum	--	<input type="radio"/> N <input checked="" type="radio"/> Y	
Delete_priv	enum	--	<input type="radio"/> N <input checked="" type="radio"/> Y	
Create_priv	enum	--	<input type="radio"/> N <input checked="" type="radio"/> Y	
Drop_priv	enum	--	<input checked="" type="radio"/> N <input type="radio"/> Y	
Grant_priv	enum	--	<input type="radio"/> N <input checked="" type="radio"/> Y	
References_priv	enum	--	<input type="radio"/> N <input checked="" type="radio"/> Y	
Index_priv	enum	--	<input type="radio"/> N <input checked="" type="radio"/> Y	
Alter_priv	enum	--	<input type="radio"/> N <input checked="" type="radio"/> Y	

**Hinweis:**

Globale Berechtigungen gelten für alle Datenbanken auf einem Server.

8.4.3 Bearbeiten einer MySQL-Datenbank mit PHP

PHP stellt verschiedene Funktionen zur Arbeit mit MariaDB zur Verfügung:

PHP-Funktion	Beschreibung	Beispiel
mysqli_connect()	Stellt eine Verbindung zum Datenbankserver her.	\$erg=mysqli_connect (\$host, \$user, \$password);
mysqli_db_query()	Schickt eine SQL-Abfrage an den Datenbankserver.	\$erg=mysqli_db_query ('faradiso', 'select * from kunden');
mysqli_num_rows()	Liefert die Anzahl der Datensätze einer Abfrage zurück.	\$anzahl= mysqli_num_rows(\$erg);
mysqli_num_fields()	Liefert die Anzahl der Felder eines Datensatzes der Abfrage.	\$anzahl= mysqli_num_fields(\$erg);
mysqli_close()	Schließt eine Verbindung zum Datenbankserver.	mysqli_close();
mysqli_fetch_array()	Liefert einen Datensatz als Array.	\$liste=mysqli_fetch_array (\$ergebnis, \$typ)

Eine Verbindung zum Datenbankserver herzustellen und Daten abzufragen, erfolgt in mehreren Schritten:

1. Anmeldung des Benutzers am Datenbankserver
2. Festlegen der Datenbank
3. Absenden der Abfrage an den Datenbankserver, speichern des Ergebnisses in einer Variablen
4. Schließen der Verbindung

Die Anmeldung des Benutzers am Datenbankserver geschieht mit der Funktion `mysqli_connect ($host, $user, $passwd)`.

Eine SQL-Abfrage wird mit der Funktion `mysqli_db_query ($db, $sql)` abgeschickt. Die Funktion `mysqli_num_rows ($erg_sql)` liest z. B. die Anzahl der Datensätze einer Tabelle aus und speichert sie in der Variablen `$anz`. Die Funktion `mysqli_close ()` trennt die Verbindung zum Datenbankserver.

Das Skript zur Ausgabe der Anzahl der Datensätze lautet:

```
<?
$db = mysqli_connect('localhost','root','');
$erg_sql = mysqli_db_query('faradiso', 'select * from kunden');
$anz = mysqli_num_rows($erg_sql);
echo $anz;
mysqli_close();
?>
```

Herstellen einer Verbindung zum Datenbankserver

Auslesen des Ergebnisses der SQL-Anweisung

Schließen der DB-Verbindung

Anzeigen des Ergebnisses

Absenden der SQL-Anweisung

**Beispiel:**

Es wird ein PHP-Skript entworfen, welches die Tabelle `Kunden` der Datenbank `faradiso` am Bildschirm als HTML-Tabelle ausgibt.

Das Skript baut zunächst die Verbindung auf und schickt die SQL-Anweisung ab. Die Anzahl der benötigten Spalten wird mit der PHP-Funktion `mysqli_num_fields ($erg_sql)` ermittelt und der Variablen `$anzahl` übergeben. Mit einer `for`-Schleife wird der Tabellenkopf erzeugt, der mittels der Funktion `mysqli_field_name ($erg_sql, $i)` mit den Feldnamen beschriftet wird.

Innerhalb einer `While`-Schleife wird mithilfe der Funktion `mysqli_fetch_array ($erg_sql, MYSQL_ASSOC)` bei jedem Durchlauf ein Datensatz des Abfrageergebnisses in ein assoziatives Array `$zeile` eingelesen. Eine `Foreach`-Schleife ermöglicht