Zugriff auf eine MySQL-Datenbank aus einem PHP-Script

Die Daten einer bereits existierenden Datenbank können ausgewertet werden, indem folgende Schritte nacheinander durchgeführt werden. Hier im Beispiel arbeiten wir mit einer Datenbank, die u.a. eine Tabelle mit Adressen enthält.

1. Herstellen einer Verbindung zum MySQL-Server und zur gewünschten Datenbank (mit `mysqli_connect` und Zeichensatz festlegen).
2. Abfrage als SQL-Anweisung an die Datenbank senden (mit `mysqli_query`)
3. Auswerten der Ergebnistabelle (mit `mysqli_fetch_object(), mysqli_fetch_row()`)
4. Schließen der DB und der DB-Verbindung (mit `mysqli_close`).

1. Herstellen der Verbindung zum MySQL-Server und zur Datenbank:

Die PHP-Anweisung `mysqli_connect(...)` erhält folgende Parameter: Die Adresse des DB-Servers, den Benutzernamen, das Passwort und den Datenbanknamen.

```
$db_link = mysqli_connect("localhost", "Benutzername", "Passwort",
"Datenbankname");
```

Sie liefert in `$db_link` eine Verbindung zurück, über die man auf die Datenbank zugreifen kann.

Beispiel.:

```
$db_link=mysqli_connect("localhost","root","","xyz" );
```

Hier liegt der MySQL-Server (DB-Server) auf Ihrem lokalen Rechner, also `"localhost"`. Die Verbindung zum DB-Server und zur DB wird als Benutzer `"root"` ohne Passwort `""` durchgeführt und zum Zugriff auf Ihre DB muss der **Name Ihrer Datenbank** (im Bsp „xyz“) angegeben werden. Liegt der Server auf einem anderen Rechner, muss anstelle von localhost dessen IP-Adresse angegeben werden.

Bevor Anfragen an die Datenbank gestellt werden, muss überprüft werden, ob die Verbindung zur DB erfolgreich war. Im Fehlerfall erfolgt eine entsprechende Fehler-Meldung (s. Bsp.).

```
if($db_link){ ..Anweisungsblock.. } else echo "Keine Verbindung möglich";
```

Es ist sinnvoll anzugeben, mit welchem Zeichensatz die Datenbank arbeiten soll (sonst werden die Umlaute „öäü“ nicht korrekt dargestellt).

```
mysqli_set_charset($db_link, 'utf8');
```

(UTF-8 ist eine 8-Bit-Zeichencodierung für Unicode. Die Abkürzung „UTF-8“ steht für „8-Bit Universal Character Set Transformation Format“.)

2. Abfragen an die Datenbank

Abfragen an die DB werden mit der Anweisung `mysqli_query($db_link,$SQLString)` gesendet. Der Parameter `$db_link` enthält die Datenbankverbindung. Der Parameter `$SQLString` enthält den String mit der SQL-Anweisung zum Suchen der Daten in der DB. Die Anweisung liefert eine spezielle Ergebnistabelle zurück, die in der Variable `$Ergebnis` gespeichert ist.

Beispiel:

```
$SQLString="SELECT * FROM Adresse"; //Alle Einträge der Tabelle Adresse werden selektiert
$ergebnis=mysqli_query($db_link,$SQLString);
```

Es ist übersichtlich, die SQL-Anweisung in einen String zu schreiben und mittels `echo` im Browser auszugeben. Der SQL-String kann dann unabhängig vom Skript auf der DB überprüft werden.

Es sollte weiterhin geprüft werden, ob ein gültiges (korrektes) Ergebnis erhalten wurde:

```
if (!$ergebnis) { //Fehlermeldung } else { ..Anweisungsblock.. }
```

3. Auswerten des Abfrage-Ergebnisses (`mysqli_fetch_object()` und `mysqli_fetch_row()`)

Die Anweiswung `mysqli_query()` liefert im folgenden Beispiel eine Ergebnistabelle `$ergebnis` mit zwei Spalten, die jeweils Vornamen und Nachnamen aus der Adressen-Tabelle enthalten. Enthält das Ergebnis nur eine einzige Zeile, kann diese eine Zeile direkt aus dem `$ergebnis` mit `mysqli_fetch_object()` oder `mysqli_fetch_row()` ohne Verwendung einer Schleife ausgelesen werden.

Beispiel: `$row = mysqli_fetch_object($ergebnis);`

- `mysqli_fetch_object()`

Die Daten werden zeilenweise in einer Schleife ausgegeben. Mit `mysqli_fetch_object()` wird jeweils eine Zeile gelesen und in der Variable `$row` abgespeichert. Mit der Variable `$row` wird „objektorientiert“ auf die einzelnen Elemente in der Zeile (`$row`) zugegriffen:

```
$ergebnis = mysqli_query($db_link, "SELECT Vorname, Name FROM Adresse");
while($row = mysqli_fetch_object($ergebnis))
{
  echo $row->Vorname;     //Vorname wird aus der Zeile $row geholt
  echo $row->Name;        //Name wird aus der Zeile $row geholt
  echo "<br>";
}
```

- `mysqli_fetch_row()` erlaubt den Zugriff auf einzelne Spalten der Ergebnistabelle.

Die Daten werden zeilenweise in einer Schleife ausgegeben. Mit `mysqli_fetch_row()` wird jeweils eine Zeile gelesen und in der Variable `$row` abgespeichert. Mit der Variable `$row` wird auf die einzelnen Elemente wie in einem Array der Programmiersprache C zugegriffen:

```
$ergebnis = mysqli_query($db_link, "SELECT Vorname, Name FROM Adresse");
while ($row = mysqli_fetch_row($ergebnis))
{
  echo $row[0], $row[1];     //Vorname und Name werden aus der Zeile $row geholt
  echo "<br>";
}
```

4. Schließen der DB-Verbindung

`mysqli_close()` schließt eine Verbindung zur DB. Die Verbindung in `$db_link`, die durch die PHP-Anweisung `mysqli_connect` erhalten wurde, wird als Parameter angegeben.

Beispiel: `mysqli_close($db_link);`

Weitere Funktionen, um die Ergebnistabelle (das Abfrage-Ergebnis) auszuwerten:

- `$rowcount=mysqli_num_rows($Ergebnis);` liefert die Anzahl der Zeilen der Ergebnistabelle. Wird 0 geliefert, wurde nichts gefunden.
- `$fieldcount=mysqli_num_fields($Ergebnis);` liefert die Anzahl der Spalten der Ergebnistabelle.
- `mysqli_affected_rows($db_link)` liefert die Anzahl der geänderten Datensätze. Damit kann z.B. nach einer UPDATE- oder DELETE-SQL-Anweisung geprüft werden, wie viele und ob überhaupt Datensätze geändert wurden. Hier im Beispiel werden in der Adressentabelle alle Adressen gelöscht, die als Name 'Maier' haben.

```
$SQLString="DELETE FROM Adresse WHERE Name='Maier'";
$Ergebnis=mysqli_query($db_link,$SQLString);
echo "Gelöschte Datensätze: ", mysqli_affected_rows($db_link);
```