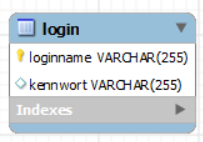
# L2\_2.2 Daten aus einer Datenbank auslesen und auswerten inklusive einer Fehlerprüfung

Der programmiertechnische Zugriff auf die Daten eines DBMS ist mit wenigen Zeilen Programmcode zu bewältigen. Allerdings wurden im Eingangsbeispiel keinerlei Fehlerprüfungen unternommen. Dabei kann beim Ansprechen des DBMS vieles schief gehen. So könnte das DBMS gar nicht laufen oder sich auf einem anderen als den angegebenen Server befinden. Weiter besteht die Gefahr, dass die von uns formulierter SQL-Anweisung fehlerhaft ist und vom DBMS nicht ausgeführt werden kann. Daher wird im Folgenden eine weitere Abfrage ausgeführt, allerdings mit einer umfangreicheren Fehlerprüfung.

In diesem Beispiel wollen wir prüfen, ob die Zugangsdaten – also der Anmeldename und das Kennwort – eines Benutzers korrekt sind. Dazu werden diese Zugangsdaten in einer Tabelle mit folgender Struktur gespeichert:



Das Attribut „loginname“ bildet den Primärschlüssel und beinhaltet den Anmeldename. Das Kennwort findet sich im gleichnamigen im Attribut.

Zur Eingabe der Zugangsdaten steht bereits ein HTML-Formular zur Verfügung:

|  |  |
| --- | --- |
| Die für die Auswertung in der PHP-Datei wichtigen Informationen sind rot umrahmt! | […]  <form action="loginpruefen.php" method="POST">    <fieldset>      <label for="name">Anmeldename</label>      <input type="text" id="name" name="name">      <label for="kw">Kennwort</label>      <input type="password" id="kw" name="kennwort">      <input type="submit" value="Anmelden">    </fieldset>  </form>  […] |

Ein PHP-Programm zur Ansteuerung eines DBMS enthält wieder die vier wesentlichen Grundbausteine. Allerdings prüfen wir zuvor die Vollständigkeit und Richtigkeit, der an die PHP-Datei übermittelten Daten. Fehlen diese, dann können wir uns einen „teuren“ Verbindungsaufbau sparen und gleich das Scheitern der Anmeldung bekanntgeben.

1. Validierung der relevanten Daten (hier: Anmeldename und Kennwort)
2. Der Verbindungsaufbau
3. Das Senden einer Anfrage an das DBMS
4. Die Auswertung der Antwort des DBMS
5. Der Verbindungsabbau

**0. Überprüfung der Vollständigkeit und Gültigkeit der übergebenen Parameter**

Hilfreich sind hierbei vor allem die PHP-Funktionen isset() und empty(), manchmal auch Funktionen zur Analyse einer Zeichenkette, wie beispielsweise die Kombination von strlen() und trim(), um die Länge einer Zeichenkette zu bestimmen.

|  |  |
| --- | --- |
| isset(*var*) | Prüft, ob eine Variable existiert und ob sie nicht NULL ist |
| empty(*var*) | Prüft, ob eine Variable einen Wert enthält. Gibt FALSE zurück, wenn *var* einen nicht-leeren und von 0 verschiedenen Wert hat. Folgende Dinge zählen als „nicht mit einem Wert belegt“:   * "" (eine leere Zeichenkette) * 0 (0 als Integer) * 0 (0 als Fließkommazahl) * "0" (0 als Zeichenkette) * NULL * FALSE * array() (ein leeres Array) * $var; (in einer Klasse deklarierte, aber nicht belegt Variable) |
| trim(*var*) | Entfernt Whitespaces (oder andere Zeichen) am Anfang und Ende eines Strings |
| strlen(*var*) | Liefert die Länge einer Zeichenkette |
| strpos(*var1*, *var2*) | Sucht das erste Vorkommen des Suchstrings *var2* im String *var1* und gibt die numerische Position des ersten Vorkommens zurück. |

Weitere String-Funktionen finden Sie unter <http://php.net/manual/de/ref.strings.php>.

Vor dem Verbindungsaufbau wollen wir prüfen, ob der Anmeldename und das Kennwort verfügbar sind. Zusätzlich prüfen wir, ob die Länge des Kennworts größer als drei Zeichen ist. Das Ergebnis könnte wie folgt aussehen:

$name = $\_POST['name'];

$kennwort = $\_POST['kennwort'];

if( isset($name) && !empty($name)  &&  isset($kennwort) && !empty($kennwort)  && strlen(trim($kennwort))>4  ) {

   ...

} else {

  echo "Hallo, die Zugangsdaten sind fehlerhaft.";

}

**1. Verbindungsausfbau**

Ist das DBMS nicht erreichbar oder sind die Anmeldedaten für den Zugang zum DBMS nicht korrekt, scheitert der Verbindungsaufbau. In diesem Fall wirft der Konstruktor der Klasse PDO eine Ausnahme (exception). Wie in zahlreichen anderen OO-Programmiersprachen können auch in PHP solche Ausnahmen (exceptions) über eine try-catch-Konstruktion abgefangen werden. Das im catch-Block bereitgestellte PDOException-Objekt kann weitere Informationen zur Fehlerursache liefern. Eine Beschreibung lässt sich über die Methode getMessage() ermitteln. Konkret könnte dies wie folgt aussehen:

$name = $\_POST['name'];

$kennwort = $\_POST['kennwort'];

if( isset($name) && !empty($name)  &&  isset($kennwort) && !empty($kennwort)  && strlen(trim($kennwort))>4  ) {

**try {**

        $pdo = new PDO("mysql:host=localhost;dbname=test;charset=utf8", "root", "");

...

**} catch(PDOException $ex) {**

**echo "Hallo " . $name . ", der Verbindungsaufbau ist gescheitert: " . $ex->getMessage();**

**}**

} else {

    echo "Hallo, die Zugangsdaten sind fehlerhaft.";

}

**2. Senden einer Anfrage an das DBMS**

Unter der Voraussetzung, dass der Verbindungsaufbau zum DBMS fehlerfrei funktioniert hat, wird im nächsten Schritt eine Anfrage mittels SQL formuliert und an das DBMS gesendet werden. Hierbei besteht vor allem die Gefahr, dass die SQL-Anweisung nicht korrekt ist und syntaktische Fehler beinhaltet. Im Falle eines solchen Fehlers liefert die Methode „query(…)“ des PDO-Objektes keine Ergebnismenge, sondern den Wert FALSE. Vor Auswertung der Ergebnismenge müssen wir also prüfen, ob die Abfrage überhaupt fehlerfrei verlaufen ist.

$name = $\_POST['name'];

$kennwort = $\_POST['kennwort'];

if( isset($name) && !empty($name)  &&  isset($kennwort) && !empty($kennwort)  && strlen(trim($kennwort))>4  ) {

    try {

        $pdo = new PDO("mysql:host=localhost;dbname=test;charset=utf8", "root", "");

        $sql = "SELECT loginname FROM login where loginname='$name' and kennwort='$kennwort'";

        $stm = $pdo->query($sql);

        if( $stm !== false ){

**...**

**} else {**

**echo "Hallo " . $name. ", Fehler bei der Ausführung**, Fehlercode: " . $pdo->errorCode();

**}**

    }catch(PDOException $e) {

        echo "Hallo " . $name . ", der Verbindungsaufbau ist gescheitert: " . $e->getMessage();

    }

} else {

    echo "Hallo, die Zugangsdaten sind fehlerhaft.";

}

Hinweis:

Auch wenn die auftretenden Fehlercodes im Internet recherchierbar sind (z.B. unter <https://www.ibm.com/support/knowledgecenter/SSEPEK_12.0.0/java/src/tpc/imjcc_rjvjcsqc.html>) sind diese wenige aussagekräftig. Alternativ kann auch die Methode „errorInfo()“ des PDO-Objektes aufgerufen werden, welche ein Array mit einem alphanumerischen Fehlercode, einem treiberspezifischen Code und einer treiberspezifischen Nachricht liefert. Auf die Nachricht kann somit mit folgender Anweisung zugegriffen werden:

$pdo->errorInfo()[2]

**3. Auswertung der Antwort des DBMS**

Bei der Auswertung sind in der Regel keine weiteren Prüfungen notwendig. Im konkreten Beispiel ist zu beachten, dass die Auswertung ohne eine Schleife auskommt. Die SQL-Anweisung ist so formuliert, dass genau ein Datensatz gefunden wird, wenn die Zugangsdaten gültig sind, ansonsten keiner. Somit muss lediglich geprüft werden, ob das DBMS einen Datensatz liefert. Dafür reicht eine Verzweigung aus.

4**. Verbindungsabbau**

Beim Verbindungsabbau sind keine weiteren Prüfungen notwendig.

**Zusammenfassung**

Das Ergebnis verfügt über die gleiche Funktion, wie das Eingangsbeispiel. Allerdings sorgen hier zahlreiche Prüfungen dafür, dass Fehlermeldungen nicht unkontrolliert in der Ausgabeseite erscheinen. Werden Fehler erkannt, erfolgt eine für den Anwender lesbare Fehlermeldung.

<?php

$name = $\_POST['name'];

$kennwort = $\_POST['kennwort'];

*//0. Validierung der notwendigen Daten*

if( isset($name) && !empty($name)  &&  isset($kennwort) && !empty($kennwort)  && strlen(trim($kennwort))>3  ) {

    try {

*//1. Verbindungsaufbau*

        $pdo = new PDO("mysql:host=localhost;dbname=test;charset=utf8", "root", "");

*//2. Formulieren und Senden der Anfrage*

        $sql = "SELECT loginname FROM login2 where loginname='$name' and kennwort='$kennwort'";

        $stm = $pdo->query($sql);

        if( $stm !== false ){

*//3. Auswertung der DB-Antwort*

            if ($row = $stm->fetchObject()) {

                echo "Hallo " . $row->loginname . ", Sie sind nun angemeldet";

            } else {

                echo "Hallo " . $row->loginname . ", die Zugangsdaten sind leider nicht korrekt.";

            }

*//4. Verbindungsabbau*

            $stm = null;

            $pdo = null;

        } else {

            echo "Hallo " . $name . ", Fehler bei der Ausführung, Fehlercode: " . $pdo->errorCode();

        }

    }catch(PDOException $e) {

        echo "Hallo " . $name . ", der Verbindungsaufbau ist gescheitert: " . $e->getMessage();

    }

} else {

    echo "Hallo, die Zugangsdaten sind fehlerhaft.";

}

?>