**Lektion 20: Daten aus einer Datenbank abrufen**

Jetzt ist es Zeit, Daten aus unserer Datenbank abzurufen.

Dies ist wirklich eine der wichtigsten Lektionen dieses Tutorials. Wenn Sie diese Lektion bearbeitet und verstanden haben, werden Sie erkennen, warum datenbankgestützte Weblösungen so mächtig sind und Ihre Sichtweise auf Webentwicklung wird sich drastisch verändern.

**SQL-Queries**

Um Daten aus einer Datenbank abzurufen, benutzen sie sog. "Queries" oder Abfragen. Ein Beispiel für eine Abfrage wäre z.B. "gib alle Daten aus der Tabelle 'Personen' aus" oder "gib die Namen aus der Tabelle 'Personen' alphabetisch sortiert aus".

Nochmal, die 'Sprache' **S**tructured **Q**uery **L**anguage (SQL) wird genutzt, um mit der Datenbank zu kommunizieren. Schauen Sie sich einmal dieses einfache Beispiel an:

Gib alle Daten aus der Tabelle 'Personen' aus

würde in SQL so geschrieben:

SELECT \* FROM Personen

Die Syntax ist relativ selbsterklärend. Lesen Sie weiter und sehen Sie, wie die SQL-Anweisungen in den folgenden Beispielen benutzt werden.

**Beispiel 1: Daten aus einer Tabelle abrufen**

Dieses Beispiel benutzt die Datenbank und Tabelle aus [Lektion 19](http://de.html.net/tutorials/php/lesson19.php) und [Lektion 18](http://de.html.net/tutorials/php/lesson18.php). Aus diesem Grund ist es wichtig, dass Sie sich zuerst diese Lektionen durchlesen.

Das Beispiel zeigt, wie Daten aus der Tabelle 'Personen' mit einer SQL-Query abgefragt werden.

Die SQL-Query gibt ein Resultat in Form einer Reihe von **Datensätzen** (record) zurück. Diese Datensätze werden in sog. **Datensatzgruppen** (recordset) gespeichert. Eine Datensatzgruppe kann man sich am ehesten als eine Art Tabelle im Speicher der Servers vorstellen, die Zeilen von Daten (die Datensätze) enthält. Jeder Datensatz ist wiederum in individuelle Felder (oder Spalten) unterteilt.

Eine Datensatzgruppe ist vergleichbar mit einer Tabelle und ein Datensatz mit einer Zeile in der Tabelle. In PHP können wir mit einer Schleife und der Function [[Dokumentation](http://php.net/manual/en/function.mysql-fetch-array.php)mysql\_fetch\_array](http://php.net/manual/en/function.mysql-fetch-array.php) durch die Datensatzgruppe gehen. Letztere gibt jede Zeile als Array aus.

Der folgende Code zeigt, wie [[Dokumentation](http://php.net/manual/en/function.mysql-fetch-array.php)mysql\_fetch\_array](http://php.net/manual/en/function.mysql-fetch-array.php) bei den Schleifendurchläufen durch eine Datensatzgruppe benutzt wird:

<html>

<head>

<title>Daten aus einer Datenbank abrufen</title>

</head>

<body>

<?php

// Verbindung zum Datenbankserver

mysql\_connect("mysql.meinserver.de", "benutzer001", "m31np455w0rt") or die (mysql\_error ());

// Datenbank auswählen

mysql\_select\_db("meinedatenbank") or die(mysql\_error());

// SQL-Query

$strSQL = "SELECT \* FROM Personen";

// Query ausführen (die Datensatzgruppe $rs enthält das Ergebnis)

$rs = mysql\_query($strSQL);

// Schleifendurchlauf durch $rs

// Jede Zeile wird zu einem Array ($row), mit mysql\_fetch\_array

while($row = mysql\_fetch\_array($rs)) {

// Schreibe den Wert der Spalte Vorname (der jetzt im Array $row ist)

echo $row['Vorname'] . "<br />";

}

// Schließt die Datenbankverbindung

mysql\_close();

?>

</body>

</html>

 - [Beispiel anzeigen](http://de.html.net/tutorials/php/lesson20_ex1.php)

Aus jedem Datensatz können wir jetzt den Inhalt der Spalte "Vorname" auslesen indem wir $row['Vorname'] schreiben. Ähnlich verfahren wird mit den anderen Spalten, z.B. der Spalte "Telefon", $row['Telefon'].

Die Reihenfolge der Datensatzgruppe ist genau die gleiche wie in der Tabelle in der Datenbank. Aber im nächsten Beispiel zeigen wir, wie man eine Datensatzgruppe sortiert.

**Beispiel 2: Daten alphabetisch, chronologisch oder numerisch sortieren**

Es ist oft hilfreich, wenn eine Liste oder Tabelle von Daten dem Alphabet nach, chronologisch oder numerisch sortiert ist. Eine solche Sortierung ist mit SQL sehr einfach zu bewerkstelligen, wobei die Syntax **Order By Spaltenname** für das Sortieren nach dem Inhalt der entsprechenden Spalte genutzt wird.

Sehen Sie sich nochmal die SQL-Anweisung aus dem Beispiel oben an:

strSQL = "SELECT \* FROM Personen"

Die Datensätze können, zum Beispiel, alphabetisch nach dem Vornamen der Personen sortiert werden. Und zwar so:

strSQL = "SELECT \* FROM Personen ORDER BY Vorname"

Oder chronologisch nach dem Geburtsdatum:

strSQL = "SELECT \* FROM Personen ORDER BY Geburtsdatum"

Die Sortierung kann von **aufsteigend** (ascending) zu **absteigend** (descending) geändert werden, indem man **DESC** hinzufügt:

strSQL = "SELECT \* FROM Personen ORDER BY Geburtsdatum DESC"

Im folgenden Beispiel werden die Personen nach dem Alter sortiert:

<html>

<head>

<title>Daten aus einer Datenbank abrufen</title>

</head>

<body>

<?php

// Verbindung zum Datenbankserver

mysql\_connect("mysql.meinserver.de", "benutzer001", "m31np455w0rt") or die (mysql\_error ());

// Datenbank auswählen

mysql\_select\_db("meinedatenbank") or die(mysql\_error());

// SQL-Query

$strSQL = "SELECT \* FROM Personen ORDER BY Geburtsdatum DESC";

// Query ausführen (die Datensatzgruppe $rs enthält das Ergebnis)

$rs = mysql\_query($strSQL);

// Schleifendurchlauf durch $rs

while($row = mysql\_fetch\_array($rs)) {

// Schreibe den Wert der Spalten Vorname und Geburtsdatum

echo $row['FirstName'] . " " . $row['BirthDate'] . "<br />";

}

// Schließt die Datenbankverbindung

mysql\_close();

?>

</body>

</html>

 - [Beispiel anzeigen](http://de.html.net/tutorials/php/lesson20_ex2.php)

Versuchen Sie, die SQL-Anweisung selbst zu verändern und die Datensätze nach Vorname, Name oder Telefonnummer ordnen zu lassen.

**Ausgewählte Daten ausgeben**

Bisher liefert unsere SQL-Anweisung immer **alle** Zeilen einer Tabelle. Oft müssen Sie in der Anweisung aber ein Kriterium festlegen, welche Daten gesucht werden sollen. Sie suchen z.B. nur nach einer bestimmten Telefonnummer oder einem bestimmten Namen.

Nehmen wir an, wir wollen nur Personen suchen, deren Telefonnummer "66554433" ist. Dies könnte man so machen:

strSQL = "SELECT \* FROM Personen WHERE Telefon = '66554433 '"

Es gibt sechs relationale Operatoren in SQL:

**=** - ist gleich  
**<** - kleiner als  
**>** - größer als  
**<=** - kleiner-gleich  
**>=** - größer-gleich  
**!=** - ungleich

Zudem gibt es einige logische Operatoren:

**AND** - UND  
**OR** - ODER  
**NOT** - NICHT

Schauen Sie in [Lektion 6](http://de.html.net/tutorials/php/lesson6.php) nach, wenn Sie weitere Informationen über das Festlegen von Bedingungen benötigen.

Im nächsten Beispiel benutzen wir Bedingungen um ein Adressbuch zu erstellen.

**Beispiel 3: Adressbuch**

In diesem Beispiel werden wir versuchen, viele der Dinge, die Sie bereits gelernt haben, zu kombinieren. Wir werden eine Namensliste aus der Datenbank generieren, wobei jeder Name ein Verweis auf weitere Details zu jener Person ist.

Dafür benötigen wir zwei Dateien - **list.php** und **person.php** - mit dem folgenden Code:

**Der Code von list.php**

<html>

<head>

<title>Daten aus einer Datenbank abrufen</title>

</head>

<body>

<ul>

<?php

// Verbindung zum Datenbankserver

mysql\_connect("mysql.meinserver.de", "benutzer001", "m31np455w0rt") or die (mysql\_error ());

// Datenbank auswählen

mysql\_select\_db("meinedatenbank") or die(mysql\_error());

// SQL-Query

$strSQL = "SELECT \* FROM Personen ORDER BY Vorname DESC";

// Query ausführen (die Datensatzgruppe $rs enthält das Ergebnis)

$rs = mysql\_query($strSQL);

// Schleifendurchlauf durch $rs

while($row = mysql\_fetch\_array($rs)) {

// Name der Person

$strName = $row['FirstName'] . " " . $row['LastName'];

// Erstelle einen Link zu person.php mit dem id-Wert in der URL

$strLink = "<a href = 'person.php?id = " . $row['id'] . "'>" . $strNavn . "</a>";

// Link in der Liste

echo "<li>" . $strLink . "</li>";

}

// Schließt die Datenbankverbindung

mysql\_close();

?>

</ul>

</body>

</html>

**Der Code von person.php**

<html>

<head>

<title>Daten aus einer Datenbank abrufen</title>

</head>

<body>

<dl>

<?php

// Verbindung zum Datenbankserver

mysql\_connect("mysql.meinserver.de", "benutzer001", "m31np455w0rt") or die (mysql\_error ());

// Datenbank auswählen

mysql\_select\_db("meinedatenbank") or die(mysql\_error());

// Daten aus der Datenbank abrufen, wobei der Wert von id aus in der URL berücksichtigt wird

$strSQL = "SELECT \* FROM Personen WHERE id=" . $\_GET["id"];

$rs = mysql\_query($strSQL);

// Schleifendurchlauf durch $rs

while($row = mysql\_fetch\_array($rs)) {

// Schreibe die Daten der Person

echo "<dt>Name:</dt><dd>" . $row["Vorname"] . " " . $row["Name"] . "</dd>";

echo "<dt>Telefon:</dt><dd>" . $row["Telefon"] . "</dd>";

echo "<dt>Geburtsdatum:</dt><dd>" . $row["Geburtsdatum"] . "</dd>";

}

// Schließt die Datenbankverbindung

mysql\_close();

?>

</dl>

<p><a href="list.php">Zurück zur Liste</a></p>

</body>

</html>

 - [Beispiel anzeigen](http://de.html.net/tutorials/php/list.php)

Das Adressbuch-Beispiel ist relativ einfach, aber es zeigt das Potential des Arbeitens mit PHP und Datenbanken.

Stellen Sie sich vor, die Datenbank enthielte 10.000 Produkte mit detaillierten Beschreibungen. Wenn man die obigen Dateien ein klein wenig abändert, können Sie daraus einen Produktkatalog mit mehr als 10.000 Seiten machen - **mit nur einer Datenbank und zwei PHP-Dateien**.

Wilkommen in der Welt von umfangreichen Webseiten, die einfach zu entwickeln und zu warten sind! Wenn Sie einmal gelernt haben, mit Datenbanken zu arbeiten, werden Ihre Weblösungen nie wieder die gleichen sein.