PHP, Schüler mit Datenbank Teil 2b
Sicherheit - Prepared Statements (mit Platzhalter arbeiten)

1)Erklärung

Um SQL-Injections zu verhindern empfiehlt sich der Einsatz von **prepared statements**. Sobald man irgendwelche Daten vom Benutzer an die Datenbank übergibt, sollte man stets auf prepared Statements zurückgreifen. **Mittels prepared Statements ist man gegen SQL-Injections geschützt.**

SQL-Injections:

Ein Angreifer kann über den GET-Parameter die SQL-Abfrage manipulieren und weiteren SQL-Code einschleusen. Im schlimmsten Fall werden dadurch sensible Daten ausgegeben, Tabelle verändert oder gar ganze Tabellen gelöscht.

Bei „prepared statements“ wird ein **Platzhalter** anstelle von dem aktuellen Parameter verwendet. Dieser Platzhalter wird durch den aktuellen Wert ersetzt, wenn das Statement ausgeführt (executed) wird.

Es besteht aus zwei Stufen, dem „prepare“ und dem „execute“.

Unterschied ob man MySQLi oder PDO benutzt:

MySQLi supports the use of anonymous positional placeholder (`?`), as shown below:

```
INSERT INTO persons (first_name, last_name, email) VALUES (?, ?, ?);
```

While, PDO supports both anonymous positional placeholder (`?`), as well as the named placeholders. A named placeholder begins with a colon (`:`) followed by an identifier, like this:

```
INSERT INTO persons (first_name, last_name, email)
VALUES (:first_name, :last_name, :email);
```

Beispiel mit MySQLi:

https://www.tutorialrepublic.com/php-tutorial/php-mysql-prepared-statements.php

Tipps:

Im Prinzip sind 3 Schritte durchzuführen:

1. prepare
2. bind_param
3. execute

2)Öffne die Datei „registrieren2.php“:

- im INSERT INTO –Befehl muss man in den VALUES die Variablen durch Platzhalter ersetzen – dabei muss man genau die selbe Anzahl beachten

```
19  $sql = "INSERT INTO user (vorname, nachname, geschlecht, geb, email, passwort, bild)
20      VALUES ('$vorname','$nachname','$geschlecht','$geb','$email','$pw','$bild')";
```

wird zu

```
19  $sql = "INSERT INTO user (vorname, nachname, geschlecht, geb, email,
20      passwort, bild) VALUES (?,?,?,?,?,?,?)";
21  //    7 Platzhalter
```

- die Variable „ergebnis“ wird entfernt und durch die Variable „stmt“ ersetzt. (Sehr häufig findet man diese auch im vollen Wortlaut, nämlich „statement“. Kann man machen wie man will.)
  Dies Variable „$stmt“ bezieht sich in den Argumenten auf die bereits bekannten Variablen $con und $sql. Diese erhalten die php-bekannte Funktion „prepare“

```
23  //$ergebnis = $con->query ($sql)
24  //  or die("Fehler bei der Datenbankabfrage.");
25
26  $stmt = mysqli_prepare($con, $sql);
```

- Danach werden die oben eingefügten Parameter den vorhandenen Variablen nun zugeordnet. Das erfolgt mit der Methode „bind_param()“.
  Dabei steht das „s“ für „string“.

```
27
28  //soviele Platzhalter s wie Elemente
29  mysqli_stmt_bind_param($stmt, "sssssss", $vorname, $nachname,
30  $geschlecht, $geb, $email, $pw, $bild);
```

- Darunter wird dann diese Methode „exekutiert“, also ausgeführt.

```
31
32  mysqli_stmt_execute($stmt);
33
34  //    schöne Ausgabe
35  echo "Danke für die Meldung, es handelt sich um $vorname $nachname.";
```

Damit auch bei dieser Methode das Bild perfekt übernommen wird und in die Datenbank übernommen werden kann, muss man eine kleine Änderung vornehmen.

Entferne die PHP addslashes() – Funktion. Beachte, dass auch die Beginn- und End-Klammer entfernt werden:

```
16  $bild = addslashes(file_get_contents($_FILES['bild']['tmp_name']));
```

wird zu

```
16  $bild = file_get_contents($_FILES['bild']['tmp_name']);
```

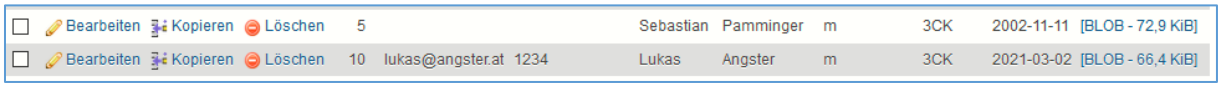
Info: Die addslashes-Funktion addiert einen Backslash bei jedem doppelten Anführungszeichenm um diese zu maskieren. Das stört jedoch bei den prepared statements.
https://www.w3schools.com/php/func_string_addslashes.asp

Problem gelöst: Das Bild soll auch angezeigt werden.

Das Bild hier wird nicht angezeigt:

5	Sebastian	Pamminger	m	3CK	2002-11-11	
10	Lukas	Angster	m	3CK	2021-03-02	

Obwohl es in der Datenbank angekommen ist:



☐	Bearbeiten	Kopieren	Löschen	5			Sebastian	Pamminger	m	3CK	2002-11-11	[BLOB - 72,9 KiB]
☐	Bearbeiten	Kopieren	Löschen	10	lukas@angster.at	1234	Lukas	Angster	m	3CK	2021-03-02	[BLOB - 66,4 KiB]

Lösung: (2022)

Das „addslashes“ gehört entfernt:

```
$bild = file_get_contents($_FILES['bild']['tmp_name']);
```

https://stackoverflow.com/questions/27516751/mysqli-prepare-statement-breaks-image-insert-into-database

> I needed the addslashes function in the old mysql statement but not when it is prepared now. Making it `$image = file_get_contents($_FILES['inputPic']['tmp_name'])` resolved the issue