Lektion 5 – HTML, CSS, PHP och MySQL

I den här lektionen behandlas i huvudsak PHP för att läsa information från en databas, MySQL. Det förutsätts att tidigare lektioner är gjorda, eller att du har tillräckliga kunskaper i grundläggande HTML, CSS och PHP för att klara den här lektionen.

Det första ni ska göra innan ni börjar arbetet med lektionen är att fylla i informationen i en databas. Databasen kommer så småningom att innehålla ett antal poster och ni kommer under lektionen att göra olika typer av utläsningar av denna data. Lösenordet är: tnmk30

www.itn.liu.se/~nikro27/tnmk30databas/

I denna lektion kommer PHP att användas för att läsa information från en databas och sedan visa innehållet på en webbsida.

Börja så här

1. Fyll i databasen och submitta informationen. Använd tnmk30 som lösenord.
2. HTML-sidan med formuläret på anropar en PHP-sida (`action`) som körs på webbservern med metoden `post`.
3. PHP-sidan börjar med att öppna en koppling till databasen med `mysqli_connect`. I anropet till databasen anges sökvägen till databasen, användare, lösenord, och databasens namn.

```
$connection = mysqli_connect("url","user","password","database");
```

4. Därefter kontrolleras att lösenordet som angavs i formuläret är korrekt, om inte så sker ingenting.
5. Om lösenordet är korrekt så ska information från formuläret hämtas. Först kontrolleras att POST-metoden används, `$_SERVER["REQUEST_METHOD"] == "POST"`. Sedan hämtas informationen från HTML-formuläret och varje input i formuläret sparas i en PHP-variabel:

```
$gender = mysqli_real_escape_string($_POST["gender"]);
```

   I detta exempel hämtas informationen i formulärelementet med namn `gender`, med hjälp av `POST`-metoden till PHP-variabel `$gender`.
6. Med hjälp av `real_escape_string("strängnamn")` tas specialtecken bort från strängen. Dessa specialtecken skulle kunna innebära att strängen är ett kommando till databasen (till exempel för att läsa ut viktig och hemlig information) och inte en inmatning av en sträng information.
   Läs mer här:
   http://php.net/manual/en/mysqli.real-escape-string.php
   http://php.net/manual/en/function.mysql-real-escape-string.php
7. Efter att alla poster i HTML-formuläret sparats i PHP-variabler skapas sedan en sträng som består av allt inklusive MySQL-kommandon. De MySQL-kommandon som används är `INSERT INTO` vilket gör att information, `VALUES`, läggs till i databasen i en databas.

```
$query = "INSERT INTO databaseName VALUES('$gender', '$year',
'$schoolsubject', '$unisubject', '$favworkchild', '$favworknow',
'$thing1', '$thing2', '$thing3', '$color', '$hand', '$transport',
'$listento','$movement')";
```

Från ovanstående kodbit framgår vad de olika posterna kallas i databasen, vilket är information ni behöver senare när ni ska läsa ut information från databasen.
`$gender` innehåller kön vilket deklareras som Man, Kvinna, Annat
`$year` innehåller födelseår mellan 1900 och 2016
`$schoolsubject` innehåller favoritämnet i grundskolan
`$unisubject` innehåller favoritämnet på universitetet
`$favworkchild` innehåller drömyrket som barn
`$favworknow` innehåller drömyrket nu
`$thing1, $thing2, $thing3` innehåller de tre sakerna att ta med till en öde ö
`$color` innehåller favoritfärgen
`$hand` innehåller information om höger- eller vänsterhänthet
`$transport` innehåller transportsättet för att ta sig till LiU
`$listento` innehåller vad personen lyssnar på
`$movement` innehåller hur mycket personen rör sig

8. Strängen `$query` ska sedan skickas till databasservern med `mysqli_query($connection, $query);`.
9. Därefter stängs kopplingen till databasen med `mysqli_close($connection);`.

Felsökning och debug i PHP

1. PHP körs ju på webbservern, och är därför ganska besvärligt att felsöka. Ett tämligen enkelt sätt att felsöka PHP är att skapa ett nytt PHP-dokument med följande i:

```
<?php
      error_reporting(E_ALL);
      ini_set("display_errors",1);
      include("minPHPfil.php");
?>
```

2. Denna kod startar upp så att eventuella felmeddelanden från webbservern visas på webbsidan, därefter inkluderas den `HTML`-kod som kan genereras utifrån `PHP`-koden som körs genom `include`.
   Läs mer här:
   http://php.net/manual/en/function.error-reporting.php
   http://www.w3schools.com/Php/func_error_reporting.asp
3. Tyvärr fungerar det inte att bara lägga in de två raderna med "`error_reporting`" och "`ini_set`" högst upp i den ursprungliga PHP-filen. Man måste göra det i en separat fil så här och sedan inkludera den fil man vill debugga. Varför det är så vet vi faktiskt inte.

Läsa all data från databasen

1. Först ska ni läsa all info i databasen och visa denna information på en webbsida. Anledningen till detta är för att ni ska kunna leta bland information efter några möjliga och roliga kopplingar att söka efter dessa kopplingar i databasen senare.
2. Skapa ett PHP-dokument med HTML, med `head`, `body`, och länka mot en `CSS`-fil. Ta till exempel grunden från laboration 2. Sätt i en rubrik, `h1`, ”`Generell fråga`

till 'tnmk30'" eller liknande. Följ det sätt och den stil ni använt på tidigare laborationer.

3. Öppna sedan, med hjälp av `mysqli_connect` en koppling till databasen. URLen till databasen ska vara mysql.itn.liu.se, användarnamnet är nikro27, inget lösenord ska anges, och databasnamnet är nikro27.

```
$connection = mysqli_connect("mysql.itn.liu.se","nikro27","","nikro27");
```

4. Om inte uppkopplingen till databasen fungerar bör anropet avslutas:

```
if (!$connection) {
      die('MySQL connection error');
}
```

I ovanstående if-sats används `!` i betydelsen icke. Så, om inte `$connection` har något värde ska försöket till uppkoppling avbrytas. Funktionen `die` markerar ett fatalt fel och avslutar exekveringen av `PHP`-filen med ett felmeddelande.
Läs mer här:
http://php.net/manual/en/function.die.php

5. Om det går bra att koppla upp mot databasen ska vi hämta all information som finns i databasen. Detta gör vi i denna lektion för att se vilka poster som finns i databasen, och för att komma på roliga frågor att ställa senare. Men, annars kan det vara dumt att be om all information som finns i en databas om databasen är väldigt stor… Allt som finns i databasen ska i alla fall läggas i en `PHP`-variabel, `$contents`, och vi frågar databasservern, `mysqli_query`, om information med `SELECT *` (vilket motsvarar allt) `FROM` namn på tabellen i databasen.

```
$contents = mysqli_query($connection, "SELECT * FROM tnmk30");
```

Läs mer här:
http://php.net/manual/en/mysqli.query.php

6. Efter att all information i databasen har hämtats ska denna presenteras i en tabell på webbsidan. Därför måste `HTML`-kod användas för att göra ett tabellelement. Det finns flera sätt att varva `HTML`-kod i `PHP`. Antingen avslutas `PHP` med `?>` och sedan skrivs `HTML` som vanligt, eller så används `echo` eller `print` i `PHP`-koden. Det är viktigt att tänka på att nästla `'` och `"` rätt när man väver in `HTML` i `PHP`. `Print` används så här:

```
print("<table>\n<tr>");
```

Vilket skapar ett tabellelement och en första tabellrad.
Läs mer här:
http://php.net/manual/en/function.print.php
http://php.net/manual/en/function.echo.php

7. Nu ska en `while`-loop användas för att hämta, `fetch`, vissa specifika fält, `field`, i det här fallet kolumnnamnen från databasen och spara denna information i en `PHP`-variabel, `$fieldinfo`.

```
while($fieldinfo = mysqli_fetch_field($contents)) {
        print("<th>". $fieldinfo->name . "</th>");
  }
```

Ovanstående `while`-loop kommer att fortsätta så länge det finns unika `field` i `$content`. `HTML`-elementet `<th>` (table head) kommer att få informationen som

finns på värdet `name` i `$fieldinfo`. Punkterna ”`.`” används för att konkatenera (sätta ihop) strängen som `PHP` använder för att generera `HTML`-koden.
Läs mer här om while-loopar:
http://www.w3schools.com/php/php_looping.asp
http://php.net/manual/en/control-structures.while.php
Och här om konkatenering av strängar:
http://php.net/manual/en/control-structures.while.php

8. När alla tabellhuvuden har fyllts av kolumnnamnen från databasen måste vi avsluta tabellraden `</tr>` med: `print "</tr>\n";`.
9. Därefter ska ytterligare en `while`-loop skrivas. I stället för fält, `field`, ska denna `while`-loop hämta information från en rad, `row`, i taget i databasen och lägga denna info i `PHP`-variabeln `$row`.

```
while($row = mysqli_fetch_row($contents)) {

}
```

10. `While`-loopen ska skapa en ny tabellrad, `<tr>` varje gång den körs med hjälp av `print`. Så här: `print "<tr>\n";`.
11. Denna `while`-loop ska också innehålla en `for`-loop. `For`-loopen ska köras från `$i=0`, och upp till antalet fält, `num_fields`, som finns i databasen enligt:

```
for($i=0; $i<mysqli_num_fields($contents); $i++) {
      print("<td>$row[$i]</td>");
}
```

Läs mer här:
http://php.net/manual/en/control-structures.for.php

12. Efter varje varv i `for`-loopen måste tabellraden avslutas `</tr>` med hjälp av `print`.
13. Därefter ska kopplingen till databasen avslutas med: `mysqli_close($connection);`.
14. `PHP`-koden ska avslutas med `?>`.
15. Tabellen ska avslutas i vanlig `HTML` med: `</table>`.
16. Och, resten av `HTML`-dokumentet ska avslutas korrekt, dvs `</body>` och `</html>`.
17. Spara `PHP`-koden och öppna sidan. Det kanske inte ser vackert ut, men ni ska nu ha en tabell med all data i.

Ställa specifika frågor till databasen

1. Nu ska ni ställa mer specifika frågor till databasen. Skapa därför ett nytt PHP-dokument, skriv i HTML-kod för `head` etc, lägg in en rubriknivå 1 `<h1>`, lägg till ett stycke text `<p>` där ni ska beskriva databasfrågan. Och, bestäm er för vad ni vill fråga databasen om. Tänk till, och fundera igenom den data ni ser på föregående webbsida. Formulera sedan frågan ni vill ställa till databasen i klartext och skriv in den i stycket med text på webbsidan. Jag bestämde mig för följande fråga: *Vilka tre saker tar personer som tyckte mest om matematik i grundskolan med sig till en öde ö, sorterat på bokstavsordning på sak 1.* Men, ni kan ställa vilken fråga ni vill här.
2. Skapa därefter grunden till en tabell `<table>` med en tabellrad `<tr>` och rätt antal kolumner med fördefinierade tabellhuvuden `<th>` i `HTML`-koden. Rätt antal kolumner beror på hur mycket data ni ska visa.

3. Skriv sedan en koppling till databasen på samma sätt som tidigare. Och bygg strängen med fråga till databasen där ni använder `SELECT` för att välja en kolumn enligt listan ovan, vill ni välja flera kolumner använder ni komma ”,” mellan dessa. Tala om från, `FROM`, vilken databas, `tnmk30`, som dessa ska hämtas från. Och specificera sedan vilket, WHERE, argument som gäller, `schoolsubject = 'Matematik'` i mitt fall, och sedan vill jag sortera, `ORDER BY`, resultatet på sak 1, `thing1`. Till detta behövs tre kolumner, en för varje sak.

```
$contents = mysqli_query($connection, "SELECT thing1, thing2, thing3
FROM tnmk30 WHERE schoolsubject='Matematik' ORDER BY thing1");
```

4. Vilka favoritfärger har de som är vänsterhänta? Till detta behövs en kolumn.

```
$contents = mysqli_query($connection, "SELECT color FROM tnmk30 WHERE
hand='Vänsterhänt' ORDER BY color");
```

5. Vad lyssnar personer på som rör sig 0-5km och vilken favoritfärg har de? Till detta behövs två kolumner.

```
$contents = mysqli_query($connection, "SELECT listento, color FROM
tnmk30 WHERE movement='0-5' ORDER BY listento");
```

6. Vilket drömyrke som barn hade kvinnorna som lyssnar mest till musik? Till detta behövs en kolumn.

```
$contents = mysqli_query($connection, "SELECT favworkchild FROM tnmk30
WHERE gender='Kvinna' AND listento='Musik' ORDER BY favworkchild");
```

Läs mer här:
http://www.w3schools.com/php/php_mysql_select.asp
https://www.tutorialspoint.com/mysql/mysql-select-query.htm

7. Använd sedan en `for`-loop för att presentera informationen i `HTML`-tabellen. Denna `for`-loop ska köras lika många gånger som det finns rader i `PHP`-variabeln `$contents`. Kom ihåg att starta tabellrader `<tr>` och tabellceller `<td>` och att avsluta dessa `</td>` respektive `</tr>` på rätt sätt.

```
for ($x = 1; $x <= $contents->num_rows; $x++) {
      $row = mysqli_fetch_array($contents);
      echo "<tr><td>",$row["thing1"],"</td><td>",$row["thing2"],"</td>
            <td>",$row["thing3"],"</td></tr>";
}
```

8. Därefter kan kopplingen till databasen avslutas `mysqli_close($connection);`, och avsluta `PHP`-koden samt `HTML`-koden.
9. Kolla att webbsidan faktiskt visar det ni frågar efter.