PHP-MYSQL- Part 2

In the first part of the lecture we saw how to store information to database.

Then you may want to output that information, so that the user can see it on the page.

The first command you will need to use is the *SELECT FROM* MySQL statement that has the following syntax:

```
1 SELECT * FROM table_name;
```

This is a basic MySQL query which will tell the script to select all the records from the *table_name* table. After the query is executed, usually you would want the result from it stored inside a variable. This can be done with the following PHP code:

```
1 <?php
2 $query = $mysqli->query("SELECT * FROM table_name");
```

The whole content of the table is now included in a PHP array with the name *$result*. Before you can output this data you should change each piece into a separate variable. There are two stages.

Now, we have to set up the loop. It will take each row of the result and print the data stored there. This way we will display all the records in the table:

```
$query = "SELECT * FROM table_name";

if ($result = $mysqli->query($query)) {

   /* fetch associative array */
   while ($row = $result->fetch_assoc()) {
      $field1name = $row["col1"];
      $field2name = $row["col2"];
      $field3name = $row["col3"];
      $field4name = $row["col4"];
```

```
    $field5name = $row["col5"];
  }

  /* free result set */
  $result->free();
}
```

You can now write a full script to output the data. In this script the data is not formatted when it is printed:

```
<?php
$username = "username";
$password = "password";
$database = "your_database";
$mysqli = new mysqli("localhost", $username, $password, $database);

$query = "SELECT * FROM table_name";
echo "<b> <center>Database Output</center> </b> <br> <br>";

if ($result = $mysqli->query($query)) {

    while ($row = $result->fetch_assoc()) {
        $field1name = $row["col1"];
        $field2name = $row["col2"];
        $field3name = $row["col3"];
        $field4name = $row["col4"];
        $field5name = $row["col5"];

        echo '<b>'.$field1name.$field2name.'</b><br />';
        echo $field5name.'<br />';
        echo $field5name.'<br />';
        echo $field5name;
    }

/*freeresultset*/
$result->free();
}
```

This outputs a list of all the values stored in the database. This will give you a very basic output which is not useful for a live website. Instead, it would be better if you could format it into a table and display the information in it.

You can also use the PHP loop to repeat the appropriate code and include it as part of a larger table. The final output is:

```
<html>
<body>
<?php
$username = "username";
$password = "password";
$database = "your_database";
$mysqli = new mysqli("localhost", $username, $password, $database);
$query = "SELECT * FROM table_name";


echo '<table border="0" cellspacing="2" cellpadding="2">
      <tr>
          <td> <font face="Arial">Value1</font> </td>
          <td> <font face="Arial">Value2</font> </td>
          <td> <font face="Arial">Value3</font> </td>
          <td> <font face="Arial">Value4</font> </td>
          <td> <font face="Arial">Value5</font> </td>
      </tr>';

if ($result = $mysqli->query($query)) {
    while ($row = $result->fetch_assoc()) {
        $field1name = $row["col1"];
        $field2name = $row["col2"];
        $field3name = $row["col3"];
        $field4name = $row["col4"];
        $field5name = $row["col5"];

        echo '<tr>
                  <td>'.$field1name.'</td>
                  <td>'.$field2name.'</td>
                  <td>'.$field3name.'</td>
                  <td>'.$field4name.'</td>
                  <td>'.$field5name.'</td>
              </tr>';
    }
    $result->free();
}
?>
</body>
</html>
```

This code will print out table content and add an extra row for each record in the database, formatting the data as it is printed. As well as showing the whole database table, PHP can be used to select individual records, or records which match certain criteria. To do this you must use a variation of the SELECT query. To display the whole table you can use:

```
1 SELECT * FROM tablename
```

If you just want to select records which have *value=1* in the *field1-name* row you would use the following query:

```
1 SELECT * FROM tablename WHERE field1-name='1'
```

In the same way you could select records based on any field in the database. You can also search in more fields by adding more:

```
1 field='value'
```

sections into the query.

For example, the following query will select all records which have *value=1* in the *field1-name* row and *value=2* in the *field2-name* row:

```
1 SELECT * FROM tablename WHERE field1-name='1' AND field2-name='2'
```