Chapter 5. PHP and Database Connection

5.1 Basic Database Concepts

A database is a collection of data that is organized so that it can be easily accessed, managed and updated.

The management of data in a database system is done by means of a general purpose software package called a database management system (DBMS). A DBMS is a combination of hardware and software that can be used to setup and monitor a database and can manage the updating and retrieval of data that has been stored in it. A DBMS should manage data with accuracy, availability, usability and resilience (instant recovery).

Some commercially available DBMS are Oracle, INGRES, Sybase, Ms-SQL etc. MySQL is also an example of open-source DBMS.

MySQL

MySQL is an open-source relational database management system (RDBMS). It is the most popular database system used with PHP. MySQL is developed, distributed, and supported by Oracle Corporation.

- The data in a MySQL database are stored in tables which consists of columns and rows.
- MySQL is a database system that runs on a server.
- MySQL is ideal for both small and large applications.
- MySQL is very fast, reliable, and easy to use database system. It uses standard SQL
- MySQL compiles on a number of platforms.

5.1 Explore PhpMyAdmin panel

phpMyAdmin is a free and open source software tool written in PHP, intended to handle the administration of MySQL over the Web. phpMyAdmin supports a wide range of operations on MySQL and MariaDB. Frequently used operations (managing databases, tables, columns, relations, indexes, users, permissions, etc) can be performed via the user interface, while you still have the ability to directly execute any SQL statement.

As a portable web application, it has become one of the most popular MySQL administration tools.

phpMyAdmin access method

Access phpMyAdmin on port 80: **http://localhost:80/phpmyadmin**

5.2 Create MySQL Database Using PHP

5.2.1. Opening Database Connection

```
// Database configuration
$host = "localhost";
$username = "root";
```

```
    $password = "";
    $database_name = "acme_lab";

    //Create a database connection
    $connection = mysqli_connect($host,$username,$password);
    if (!$connection) {
        error_log("Failed to connect to MySQLi: " .
mysqli_error($connection));
        die('Internal server error');
    }
```

5.2.2. PHP.INI file Configuration

php.ini is a configuration file which controls execution of the php program - it also control modules supported eg: mysql module which is needed to connect php code with mysql database. php.ini should configured in the way that matches with environment (production or development) where its going to used.

The PHP configuration file, php.ini, is the final and most immediate way to affect PHP's functionality. The php.ini file is read each time PHP is initialized. We can configure some of the MySQL settings from php.ini file.

mysql.default_host = hostname

The default server host to use when connecting to the database server if no other host is specified.

mysql.default_user = username

The default user name to use when connecting to the database server if no other name is specified.

mysql.default_password = password

The default password to use when connecting to the database server if no other password is specified.

magic_quotes_gpc = On

This setting escapes quotes in incoming GET/POST/COOKIE data. If you use a lot of forms which possibly submit to themselves or other forms and display form values, you may need to set this directive to On or prepare to use addslashes() on string-type data.

magic_quotes_runtime = Off

This setting escapes quotes in incoming database and text strings. Remember that SQL adds slashes to single quotes and apostrophes when storing strings and does not strip them off when returning them. If this setting is Off, you will need to use stripslashes() when outputting any type of string data from a SQL database. If magic_quotes_sybase is set to On, this must be Off.

magic_quotes_sybase = Off

This setting escapes single quotes in incoming database and text strings with Sybase-style single quotes rather than backslashes. If magic_quotes_runtime is set to On, this must be Off.

5.2.3. Closing Database Connection

The mysqli_close() function closes a previously opened database connection.

//close previously opened database connection

Syntax : `mysqli_close(<connection>);`

Example : `mysqli_close($connection);`

5.2.4. Creating a Database

Database can be created in MySQL using “create database” command. We will create a database named “acme”

// Create database

Syntax : `create database <database _name>`

Example :

```
$sql = "CREATE DATABASE acme";
$query = mysqli_query($connection,$query);
if($query){
     echo "Database created successfully";
}else{
     echo "Error creating database. Error Occured : ".mysqli_error();
}
```

5.2.5. Selecting a Database

The select_db() / mysqli_select_db() function is used to change the default database for the connection. This function is used to select a particular database to work with.

Syntax : `mysqli_select_db($connection, <database_name>);`

Example:

```
     $db_select = mysqli_select_db($connection, acme_lab);
     if (!$db_select) {
          error_log("Database selection failed: " . mysqli_error($connection));
          die('Internal server error');
     }
```

5.2.6. Creating Database Tables

The CREATE TABLE statement is used to create a table in MySQL.

We will create a table named "students", with five columns: "id", "name", "roll", "program" and "email"

```
CREATE TABLE students (
     id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
     name VARCHAR(30) NOT NULL,
     roll_no INT(10),
     program VARCHAR(50) NOT NULL,
     email VARCHAR(50),
)
```

5.3 Deleting MySQL Database and Table Using PHP

We need special privileges to delete a MySQL Database.

Deleting Database

Syntax : `Drop Database <database_name>`

Example : `$sql = Drop DATABASE  acme_lab`

Deleting Database Table

Syntax : `Drop Table <table_name>`

Example : `$sql = Drop Table  students`

5.4 Insert Data into MySQL Database

To insert data into a MySQL table, you would need to use the SQL INSERT INTO command. We will insert data into students table which was created in 5.3.6.

Syntax:

```
INSERT INTO table_name ( field1, field2, ...fieldN )
VALUES ( value1, value2,...valueN );
```

Example :

```
$sql = INSERT INTO students ( name, roll_no , program, email)
     VALUES( “Koshal Shrestha”, “14”, “Comp”,
“koshal@gmail.com” );

$query = mysqli_query($connection,$sql);
if($query){
          echo "Data inserted successfully";
}else{
     echo "Error inserting data. Error  : ".mysqli_error();
}
```

5.5 Retrieve Data From MySQL Database

The SQL SELECT command is used to fetch data from the MySQL database. You can use this command in any script like PHP.

Syntax :

```
SELECT field1, field2,...fieldN
FROM table_name1, table_name2...
[WHERE Clause]
[OFFSET M ][LIMIT N]
```

Example :

```
<table width='90%' border='1px' class = 'table' align='center'>
     <tr>
          <td><b>Id</b></td>
          <td><b>Name</td>
          <td><b>Roll Number</b></td>
          <td><b>Program</b></td></td>
          <td><b>Gender</b></td>
          <td><b>Phone Number</b></td>
          <td><b>Email</b></td>
          <td><b>Address</b></td>
     </tr>
     $fetch_sql = "select * from students order by id DESC";
          $query_exec = mysqli_query($connection,$fetch_sql);
     $rows_count  = mysqli_num_rows($query_exec);
          if($rows_count){
                    while ($row = mysqli_fetch_array($query_exec))
{
                              echo "<tr>";
                              echo "<td>".
($row['id'])."</td><td>".$row['name']."</td><td>".
$row['roll']."</td><td>".$row['program']."</td><td>".
$row['gender']."</td><td>".$row['phone']."</td><td>".
$row['email']."</td><td>".$row['address']."</td>";
                              echo "</tr>";
               }
          }else{
               echo "<tr><td colspan='9'>No data found</td></tr>";
          }
```

5.6 Releasing Memory

5.7 Updating Data into MySQL Database

we can use the SQL UPDATE statement to update records. This will modify any field value of any MySQL table.

Syntax :

```
UPDATE table_name
SET field1 = new-value1, field2 = new-value2
[WHERE Clause]
```

Example :

```
$update_query = "UPDATE students set
                                    name = '{$name}',
                                         roll = '{$roll}',
                                    where id= '{$id}'";
$query = mysqli_query($connection,$update_query);
if($query){
          echo "Success";
}else{
          echo "failure : ".mysqli_error();
}
```

5.8 Deleting Data From MySQL Database

If you want to delete a record from any MySQL table, then we can use the SQL command DELETE FROM.

Syntax : `DELETE FROM table_name [WHERE Clause]`

Example : `DELETE FROM students WHERE id=10;`

5.9 Using PHP To Backup MySQL Database

Backup and recovery describes the process of creating and storing copies of data that can be used to protect organizations against data loss. It is always good practice to take a regular backup of your database. There are several ways to take backup of your database. They are

- Using SQL Command through PHP.
- Using MySQL binary mysqldump through PHP.
- Using phpMyAdmin user interface.

Among them we will discuss about database backup **Using SQL command through PHP**.

```php
<?php
// Database configuration
$host = "localhost";
$username = "root";
$password = "";
$database_name = "acme_lab";

// Get connection object and set the charset
$conn = mysqli_connect($host, $username, $password, $database_name);
$conn->set_charset("utf8");

// Get All Table Names From the Database
$tables = array();
$sql = "SHOW TABLES";
$result = mysqli_query($conn, $sql);
while ($row = mysqli_fetch_row($result)) {
    $tables[] = $row[0];
}

$sqlScript = "";
foreach ($tables as $table) {
    // Prepare SQLscript for creating table structure
    $query = "SHOW CREATE TABLE $table";
    $result = mysqli_query($conn, $query);
    $row = mysqli_fetch_row($result);

    $sqlScript .= "\n\n" . $row[1] . ";\n\n";

    $query = "SELECT * FROM $table";
    $result = mysqli_query($conn, $query);
    $columnCount = mysqli_num_fields($result);

    // Prepare SQLscript for dumping data for each table
    for ($i = 0; $i < $columnCount; $i ++) {
        while ($row = mysqli_fetch_row($result)) {
            $sqlScript .= "INSERT INTO $table VALUES(";
            for ($j = 0; $j < $columnCount; $j ++) {
                $row[$j] = $row[$j];

                if (isset($row[$j])) {
```

```
                    $sqlScript .= '"' . $row[$j] . '"';
                } else {
                    $sqlScript .= '""';
                }
                if ($j < ($columnCount - 1)) {
                    $sqlScript .= ',';
                }
            }
            $sqlScript .= ");\n";
        }
    }
    $sqlScript .= "\n";
}

//now create file and let user to download it
if(!empty($sqlScript))
{
    // Save the SQL script to a backup file
    $backup_file_name = $database_name . '_backup_' . time() . '.sql';
    $fileHandler = fopen($backup_file_name, 'w+');  //for this
directory permission should be 777
    $number_of_lines = fwrite($fileHandler, $sqlScript);
    //echo $number_of_lines; exit;
    fclose($fileHandler);
    // Now Download the SQL backup file to the browser
    header('Content-Description: File Transfer');
    header('Content-Type: application/octet-stream');
    header('Content-Disposition: attachment; filename=' .
basename($backup_file_name));
    header('Content-Transfer-Encoding: binary');
    header('Expires: 0');
    header('Cache-Control: must-revalidate');
    header('Pragma: public');
    header('Content-Length: ' . filesize($backup_file_name));
    ob_clean();
    flush();
    readfile($backup_file_name);
    exec('rm ' . $backup_file_name);
}
?>
```