MySQL: Access Via PHP

CISC 282

March 13, 2014

phpMyAdmin: Login

http://cisc282.caslab.queensu.ca/phpmyadmin/

Use your NetID and CISC 282 password to log in
phpMyAdmin: Select DB

Clicking on this icon will log you out.

Select your database named dbNetID by clicking on it.

phpMyAdmin: New Table

Create a new table by entering its name and the number of attributes.
phpMyAdmin: New Table

Specify the data types for each field and click “Go”

At least one column must be set as the primary key

phpMyAdmin: Add Records

These tabs allow you to perform any SQL command on the database

Add new records by entering values here

All the database’s tables are listed here
phpMyAdmin: Alter Records

Click here to edit a record

Click here to delete a record

PHP and MySQL

- Prior to PHP 5.0
  - Interact with a database using `mysql_...` functions
  - These either require or return a `resource`
    - Use to access the database and execute statements

- As of PHP 5.0
  - Use the improved MySQL functionality
    - Known as MySQLi
  - What are the improvements?
    - Procedural and object-oriented approaches
    - Support for prepared statements
    - Better debugging
    - ... and more
Object-Oriented MySQLi

```php
$mysqli = new mysqli("hostName", "userName", "password", "dbName");
```

- Equivalent to ...
  - "mysql -h hostName -u userName -p password" on the command line
  - May need to use "localhost" for hostName
  - "USE dbName;" in MySQL (optional)
- Returns a MySQL object
- To connect on the CISC 282 server ...
  - use your NetID as userName
  - use your CISC 282 password as password
  - use dbName db_NetID

Plaintext Password?!?!?

- It is bad practice to store passwords in plaintext
  - Extremely unsecure should someone read the code
  - At a systems level, password files are heavily encrypted
- Option 1: obfuscate or encrypt the password
  - Will need to de-obfuscate or decrypt to use
  - What if someone gains access to the server?
    - Can reverse the process and get the values
- Option 2: set default values in a PHP config file
  - Used by the MySQLi constructor if arguments are missing
  - What if someone gains access to the server?
    - Can get the values using `phpinfo()`
    - Can log in using the defaults
Plaintext Password?!?!

- Option 3: restrict the MySQL account
  - Only permit the bare minimum for that user
  - Unable to do much damage

- What do people actually do?
  - Often, not much
  - Generally, either options 1 & 3 or 2 & 3
    - There's not a lot of consensus on the best practice
  - You don't have to deal with this for your assignments
    - But you should if you encounter this professionally

---

Error Handling

```php
$mysqli->errno;
```
- The error code from the last “fail-able” command
  - Not all function calls can fail or cause an error
  - The property is an integer value
    - 0 if no error occurred

```php
$mysqli->error
```
- The error message from the last “fail-able” command
  - The property is a string value
Error Handling

- `$mysqli->connect_errno`
- `$mysqli->connect_error`
  - Connection-specific versions of the above
  - `$mysqli->connect_error` available from PHP 5.2.9/5.3.0
    - Use `mysqli_connect_error()` instead for earlier versions

```php
die(errorMsg)
```
  - Use in combination with MySQLi error handling
  - Forces the PHP script to print an error message and exit
    - e.g.,
      ```php
      if ($mysqli->connect_errno) {
        die($mysqli->connect_error);
      }
      ```
  - Useful for debugging, but not so great for users

Connection Functions

- `$mysqli->select_db("databaseName")`
  - Equivalent to "USE databaseName;" in MySQL
  - Returns `FALSE` if the database could not be chosen
    - Returns `TRUE` otherwise
  - On the CISC 282 server, use database `dbNetID`

- `$mysqli->close()`
  - Equivalent to "quit" in MySQL
  - Closes the database connection for `$mysqli`
  - Returns `TRUE` if the connection was closed and `FALSE` otherwise