Connecting to a Database Using PHP

Prof. Jim Whitehead
CMPS 183, Spring 2006
May 15, 2006
Rationale

• Most Web applications:
  – Retrieve information from a database to alter their on-screen display
  – Store user data such as orders, tracking, address, credit card, etc. in a database

• Permits them to adapt to individual users, and provide fresh, changing content
PHP: Built-in Database Access

• PHP provides built-in database connectivity for a wide range of databases
  – MySQL, PostgreSQL, Oracle, Berkeley DB, Informix, mSQL, Lotus Notes, and more
  – Starting support for a specific database may involve PHP configuration steps

• Another advantage of using a programming language that has been designed for the creation of web apps.

• Support for each database is described in the PHP manual at:
MySQL and PHP

• Architecture diagram

<table>
<thead>
<tr>
<th>Logical Web Server</th>
<th>Logical Database (MySQL)</th>
</tr>
</thead>
<tbody>
<tr>
<td>OS proc. 1</td>
<td></td>
</tr>
<tr>
<td>Thread 1</td>
<td></td>
</tr>
<tr>
<td>Thread 2</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>Thread n</td>
<td></td>
</tr>
</tbody>
</table>

| OS proc. n         |                          |
| Thread 1           |                          |
| Thread 2           |                          |
| ...                |                          |
| Thread n           |                          |
Connecting to MySQL

• To connect to a database, need to create a connection
  – At lowest level, this is a network connection
  – Involves a login sequence (username/password)
• Since this is a relatively expensive step, web application environments:
  – Share connections
  – Have multiple connections
• Whether, and how many, are typical configuration items. In MySQL:
  – Allow_persistent: whether to allow persistent connections
  – Max_persistent: the maximum number of persistent connections
  – Max_links: max number of connections, persistent and not
  – Connection_timeout: how long the persistent connection is left open
• Can also use SSL to encrypt connection
High-Level Process of Using MySQL from PHP

• Create a database connection
• Select database you wish to use
• Perform a SQL query
• Do some processing on query results
• Close database connection
Creating Database Connection

• Use either `mysql_connect` or `mysql_pconnect` to create database connection
  – `mysql_connect`: connection is closed at end of script (end of page)
  – `mysql_pconnect`: creates persistent connection
    • connection remains even after end of the page

• Parameters
  – Server – hostname of server
  – Username – username on the database
  – Password – password on the database
  – New Link (mysql_connect only) – reuse database connection created by previous call to `mysql_connect`
  – Client Flags
    • `MYSQL_CLIENT_SSL :: Use SSL`
    • `MYSQL_CLIENT_COMPRESS :: Compress data sent to MySQL`
Security Note

• Username and password fields imply that database password is sitting there in the source code
  – If someone gains access to source code, can compromise the database
  – Servers are sometimes configured to view PHP source code when a resource is requested with “.phps” instead of “.php”
  – One approach to avoid this: put this information in Web server config. File
    • Then ensure the Web server config. file is not externally accessible
Selecting a Database

• mysql_select_db()
  – Pass it the database name

• Related:
  – mysql_list_dbs()
    • List databases available
  – Mysql_list_tables()
    • List database tables available
Perform SQL Query

- Create query string
  - $query = ‘SQL formatted string’
  - $query = ‘SELECT * FROM table’

- Submit query to database for processing
  - $result = mysql_query($query);
  - For UPDATE, DELETE, DROP, etc, returns TRUE or FALSE
  - For SELECT, SHOW, DESCRIBE or EXPLAIN, $result is an identifier for the results, and does not contain the results themselves
    - $result is called a “resource” in this case
    - A result of FALSE indicates an error

- If there is an error
  - mysql_error() returns error string from last MySQL call
Process Results

• Many functions exist to work with database results
• `mysql_num_rows()`
  – Number of rows in the result set
  – Useful for iterating over result set
• `mysql_fetch_array()`
  – Returns a result row as an array
  – Can be associative or numeric or both (default)
  – `$row = mysql_fetch_array($result);`
  – `$row[‘column name’]` :: value comes from database row with specified column name
  – `$row[0]` :: value comes from first field in result set
Process Results Loop

• Easy loop for processing results:
  ```php
  $result = mysql_query($qstring);
  $num_rows = mysql_num_rows($result);
  for ($i=0; $i<$num_rows; $i++) {
      $row = mysql_fetch_array($result);
      // take action on database results here
  }
  ```
Closing Database Connection

• mysql_close()
  – Closes database connection
  – Only works for connections opened with mysql_connect()
  – Connections opened with mysql_pconnect() ignore this call
  – Often not necessary to call this, as connections created by mysql_connect are closed at the end of the script anyway