# CSCI 5333 DBMS Classroom Notes

**11/2/2021**

URL: <http://dcm.uhcl.edu/yue/> -> IP address (Domain Name ServerL DNS)

**2. PHP**

* PHP is the acronym for PHP: Hypertext Preprocessor.
	+ Open source
	+ Multiple platforms
	+ Active community
	+ Scripting language
* <http://www.php.net/>: PHP is a widely-used general-purpose scripting language that is especially suitable for Web development and can be embedded into HTML.
* The general idea is the separation of program script code and (static) data.
* It looks a little like a mix between Perl and ASP.NET.
	+ Perl-like syntax as a scripting language.
	+ <?PHP ... ?> for inserting PHP script within HTML page.
* The PHP script section provides dynamic content for the Web pages.
* Some resources:
	+ w3schools: <http://www.w3schools.com/php/>
	+ I like: <https://learnxinyminutes.com/docs/php/>

**PHP Language Basics**

* Variables start with $.
* Important data types: <http://php.net/manual/en/language.types.php>
	1. Scalar: Boolean, number, strings
	2. Array: ordered map (of name value pairs)
	3. Object
* Comments: // or /\* \*/

URL Mapping:

**Web application in PHP:**

* Hello.php is saved in a Web account of the Web server
* When a PHP page is requested through *HTTP Request*:
	1. URL Mapping: the Web server identifies the local location of hello.php.
	2. The Web server invokes the PHP interpreter.
	3. The PHP interpreter runs hello.php.
	4. Output of hello.php is sent to the Web server (STDOUT is the Web server.)
	5. The Web server uses the output of hello.php to prepare the *HTTP response*.
* Steps 2 to 4 are governed by Common Gateway Interface (*CGI*).

URL:

<http://dcm.uhcl.edu/yue/courses/csci5333/current/notes/php/SQL_PHP.html>

DCM:



<http://dcm.uhcl.edu/yue> -> /yue/pages (root directory for the account yue)

<http://dcm.uhcl.edu/yue/courses/csci5333/current/notes/php/SQL_PHP.html> ->

 /yue/pages/ courses/csci5333/current/notes/php/SQL\_PHP.html

<http://dcm.uhcl.edu/yue/hello2.html> ->

 DCM: /yue/pages/hello2.html

Apache: default root directory: C:\xampp\htdocs

URL: localhost/php/csci5333/Fall2021/hello2.html

Local: C:\xampp\htdocs\php\csci5333\Fall2021\hello2.html





* The string using double quotes allow for *variable interpretation*. Special characters, such as $ and \ need to be escaped using \.

**phpinfo.php:** list php info. Use with care.

<?php
   phpinfo();
?>

**Server2.php:**

<table border="1">
<tr><td>Server variables</td><td>Values</td></tr>

<?php
foreach($\_SERVER as $key => $value) {
   echo "<tr><td>$key</td><td>$value</td></tr>\n";

 // the same.

   echo "<tr><td>” . $key . “</td><td>” . $value . “</td></tr>\n";
}
?>
</table>

* GET:
	1. The parameters are a part of the URL after '?', known as the *query string*.
	2. Parameters are name=value pairs, separated by the symbol '&'.
	3. The HTTP request body is empty.



Example HTTP Get Header:

1. **Request URL:**

http://dcm.uhcl.edu/yue/courses/csci5333/current/index.html

1. **Request Method:**

GET

1. **Status Code:**

200 OK

1. **Remote Address:**

129.7.84.135:80

1. **Referrer Policy:**

strict-origin-when-cross-or

GET: simplicity.

1. HTTP Header: QUerystring: name=Yue
2. NO HTTP body.

POST:

1. HTTP Header:
2. Empty line
3. HTTP Body: query string (e.g. name=yue)

Multi-part requests: multiple files; binary file; slightly more security.

**3. PHP Database Programming**

* PHP provides both Database Abstraction Layer (e.g. ODBC, JDBC: portability, diverse DB, flexibility) and Vendor-Specific Database Extensions (functionality, performance).
* Examples of Database Abstraction Layers: (DB-independent)
	+ PDO: PHP use PDO MYSQL driver to connect to a MySQL database.
	+ ODBC
	+ Pear DB: used by the text book, deprecated.
	+ Pear MDB2: <http://pear.php.net/package/MDB2>, newer version of Pear DB.
* Examples of vendor specific extensions:
	+ IBM DB2
	+ MySQL: original PHP extension for older versions; procedural interface.
	+ MySQLi: improved PHP extension for version 4.1.3 or later. Benefits from <http://php.net/manual/en/mysqli.overview.php>:
		- Object-oriented interface
		- Support for Prepared Statements
		- Support for Multiple Statements
		- Support for Transactions
		- Enhanced debugging capabilities
		- Embedded server support
	+ OCI8: Oracle OCI8
	+ PostgreSQL
	+ MS SQL Server
* *MySQLi* will be used in this class. Do not use other MySQL API for PHP.
* If you still want to use MySQL in other projects for whatever reasons: PHP MySQL function: <http://us.php.net/manual/en/ref.mysql.php>.

URL:

<http://localhost/php/csci5333/Fall2021/supply2.php?city=Houston&status=4>

* GET:
	1. The parameters are a part of the URL after '?', known as the *query string*.
	2. Parameters are name=value pairs, separated by the symbol '&'.
	3. The HTTP request body is empty.
* POST:
	1. The parameters are not a part of the URL.
	2. The HTTP request body stores the parameters.

GET: RESTFul API

<http://localhost/php/csci5333/Fall2021/get1.php?x=1&y=2>

heredoc: multiple line string literal

Variable interpretation

E.g. System Catalog: MySQL INformationSchema

**SELECT** **COUNT**(\*)

**FROM** **TABLES**

**WHERE** TABLE\_SCHEMA = 'Sakila'

**AND** TABLE\_TYPE = 'BASE TABLE';

E>g.

**SELECT** t.TABLE\_SCHEMA, **COUNT**(\*) **AS** `Num tables`

**FROM** Information\_schema.**TABLES** **AS** t

**WHERE** TABLE\_TYPE = 'BASE TABLE'

**GROUP** **BY** t.TABLE\_SCHEMA

**ORDER** **BY** `Num tables` **DESC**;



**3. Views**

* Views are *virtual* table *derived* from other tables.
* Some advantages of using views:
	1. Better data abstraction
	2. Logical data independence
	3. Better consistency
	4. Possible centralized security control
	5. Can be more efficient
* Some disadvantages:
	1. More work.
	2. Can be inefficient.
	3. Complicated especially when views are updated.
* MySQL Create View Manual: search for "mysql view manual"