INLS 760, Capra	Project 2	Spring 2017
Assigned: Feb 9	Read DB, Display Web	Due: 12:00noon, Feb 24

Overview
In this project, you will gain experience using PHP to read records from a MySQL database and then display them to a webpage by outputting HTML.

What to do:

For this project, do the following:

- In the MySQL account and database that was assigned to you on pearl.ils.unc.edu (e.g., webdb1), create a table called "p2records" with the following fields and data types:

itemnum	int
authors	varchar(200)
title	varchar(200)
publication	varchar(100)
year	char(4)
type	varchar(20)
url	varchar(255)

- Use the MySQL bulk loader (as described in class) to load records into the p2records table from the file called `bigrecords.txt` that is posted in the Resources section of the Sakai site. The format of the bigrecords.txt file will be one record per line with fields separated by vertical bars:

```
itemnum|authors|title|publication|year|type|url
```

  Each record is guaranteed to have seven fields as shown above. There will be no leading or trailing spaces. All data will be valid – you do not need to worry about checking the values. Itemnum will be a unique value for each record that can be used as the primary key for the p2records table. There is also a file called smallrecords.txt (also in the Sakai Resources) that contains records in the same format that you may find useful for testing.

- Write PHP code to query the p2records table in your database and then display the records in an HTML table with the following headings: authors, title, publication, year, type. The title should be a hyperlink to the URL for that record so that if you click on the title in a web browser, it should take you to the page specified by the URL.

- By default, your PHP code should display the records in order by item number (itemnum). However, users should be able to click any of the column headings to re-sort the records by that field. This means that information will need to be transferred from the client to the server when users click one of the column headings. There are several ways to handle this, but one simple way is to have the column headers be <a href> links that contain fields that will be sent to the PHP script using the GET method. In other words, if you have links like this:

```
<a href=”browse.php?sortby=author>Author(s)</a>
<a href=”browse.php?sortby=title>Title</a>
```

  Your PHP code should be able to read the "sortby" variable from the $_GET superglobal.

- In addition to the functionality described so far, implement a simple paging mechanism. Your PHP code should display 25 records at a time. At the bottom of the page, include links that allow the user to go forward or backward in through the records in increments of 25 records. You should provide a link for each page number. For example, if there are 180 records, you would need links for pages 1, 2, 3, 4, 5, 6, 7, 8 (where page 8 would only have 5 records). You should also include an indication of what the current page is by enclosing the current page in square brackets. In addition, the current page number should not be a hyperlink. For example, if page 4 was currently being displayed you should show the paging links like this: 1 2 3 [4] 5 6 7 8

  Sorting and paging should be able to be used together and sorting should be done before pages are determined. For example, a user should be able to view titles that start with letters at the end of the alphabet by going to the last page of results while sorting by title.

- Here is a (possibly useful) hint about how to let MySQL help with the sorting and paging:

```
SELECT * FROM p2records ORDER BY title LIMIT 100, 25
```

How to do it:
Implement the functionality described above using PHP. Be sure that your files work on opal.ils.unc.edu and the Google Chrome web browser.

Develop your solution using PHP, HTML, and CSS files as needed. **The main PHP file of your application should be named `youronyen_p2_browse.php`**. Replace `youronyen` with your Onyen. If you use other files, name them with a prefix of: `youronyen_p2_`
For example:

```
rcapra_p2_browse.php
rcapra_p2_styles.css
```

All your code and links should work with these files, named as indicated, in the same directory. In other words, if your files were placed together in some other directory, they should still work.

Note that for this and several future assignments, you will turn-in: (1) all the files you created, packaged into a tar or zip file (see below), and (2) you will also need to submit a URL to a "live" version of your project running on opal.ils.unc.edu. If you have any concerns about keeping your project running "live" on opal.ils.unc.edu, please let the instructor know.

In your tar or zip file, include a PLAIN TEXT file called `youronyen_p2_readme` that contains ONLY the URL for your "live" project, followed by a blank line. This URL should start with:

```
http://www.ils.unc.edu/~youronyen/
```

(replace `youronyen` with your Onyen). Do not include any text other than the URL and the blank line.

Grading:

The following factors will affect your grade on this assignment:

Completeness and correctness (approximately 50%)
Does your program implement all the features outlined in the assignment and do they work correctly?

Advanced Functionality (see below, 10%)

Programming implementation (approximately 30%)
Did you make good use of programming constructs such as loops, conditionals, variables, and arrays? Is your solution efficient and easy to understand?

Coding style (approximately 10%)
Is your code formatted and indented correctly? Is your code easy to understand? Did you use descriptive variable names and include comments to document and explain your approach? Do you have any lines of code that are too long (e.g., more than 100 characters long)?
For questions about coding style, I recommend: http://pear.php.net/manual/en/standards.php

What to turn in and how:
Make sure all of your files are in one directory on a Unix machine (for example, opal.ils.unc.edu). Then create a tar or zip file that contains all the files for your project. I prefer that you submit a gzipped tar file (tgz), but I will accept a regular zip file. Instructions are given below about how to create a gzipped tar file in Unix.

`cd` into the directory that contains your project files and then type the following command to create a gzipped tar file:

```
tar cvzf youronyen_p2.tgz .
```

Note that after the "tgz" there is a space and then a period. Check your tgz file with the following command:

```
tar tvzf youronyen_p2.tgz
```

If all is well, you should see a display similar to the one below.

```
[rcapra@opal]$ tar tvzf rcapra_p2.tgz
drwx------ rcapra/users      0 2007-01-17 16:57:29 ./
-rw------- rcapra/users      0 2007-01-17 16:57:29 ./rcapra_p2.tgz
-rw------- rcapra/users   4343 2007-01-17 16:48:16 ./rcapra_p2_browse.html
-rw------- rcapra/users   4343 2007-01-17 16:48:16 ./rcapra_p2_readme
-rw------- rcapra/users    115 2007-01-17 16:57:19 ./rcapra_p2_styles.css
```

When you are satisfied that your tgz file is okay, you can submit it electronically through Sakai by going to the Assignments area and finding the "P2" assignment. After you think you have submitted the assignment, I strongly recommend checking to be sure the file was uploaded correctly by downloading it from Sakai and verifying it again with the "tar tvzf" command. Keep in mind that if I cannot access your files, I cannot grade them.

If for some reason you need to re-submit your homework file, you must add a version number to your filename (for example: `youronyen_p2_v2.tgz`).

Sakai is configured with a due date and an "accept until" date. Unless you have made arrangements with the instructor in advance, submissions received after the due date may receive a 5% penalty per day.

If for some reason you are unable to submit an assignment to Sakai, as a last resort you may email it to the instructor along with a note about the problem you encountered. Then, as soon as you are able to, it is your responsibility to submit the exact same assignment to Sakai. The email serves as a record that you tried to submit the assignment on time, but to receive credit, your assignment must be uploaded to Sakai.

Advanced functionality
If you implement the program as outlined above, you can earn up to a maximum of 90 out of 100 points. You can earn up to 10 additional points for implementing the advanced feature described below.

(10 points) The authors field in the bigrecords.txt data file contains strings that include authors first names, initials, and last names (see examples below). This means that when you alphabetize based on the authors field (e.g. when you click on the authors heading), the order will be based on the first name (or initial) of the first author. For example:

```
A. Pietrinferni and S. Cassisi and M. Salaris and F. Castelli
C. Clarke and G. Lodato and S. Y. Melnikov and M. A. Ibrahimov
Swara Ravindranath and Luis C. Ho and Alexei V. Filippenko
```

Instead, we might wish to have the papers be alphabetized based on the first author's last name:

```
C. Clarke and G. Lodato and S. Y. Melnikov and M. A. Ibrahimov
A. Pietrinferni and S. Cassisi and M. Salaris and F. Castelli
Swara Ravindranath and Luis C. Ho and Alexei V. Filippenko
```

To earn points for this advanced functionality, implement code to extract the last name of the first author from the authors field. Then, when a user clicks on the authors heading to alphabetize the records by author, show the records alphabetized by the first author's last name. Hint: look for "and".