

# CS 328 - Homework 7

## Deadline

Due by 11:59 pm on **Sunday, March 27, 2016**

## How to submit

Submit your files for this homework using `~st10/328submit` on nrs-projects, with a hw number of 7.

## Purpose

To commit to a 2nd database to use for some additional applications, and to practice using PHP using OCI to connect to an Oracle database to run queries, SQL statements, and stored procedures and functions.

## Important notes

- **NOTE:** you are welcome to use `require_once` and `include_once` in your PHP documents! BUT when you use them in homework problems' documents, be sure to also SUBMIT copies of all files that you are requiring/including!
- Also note: I hope to have you present versions of some of these documents to the class at some point.
- Now there is an introductory set of **CS 328 PHP Coding Standards**, as discussed in class and as given on the public course web site -- still quite subject to change, so keep an eye on it.
- Remember to follow the **CS 328 SQL and PL/SQL Coding Standards** as given in the CS 328 Homework 2 handout and the public course web site for all SQL and PL/SQL code.
- Make sure that you have executed the scripts `create-bks.sql` and `pop-bks.sql`.
- Remember to follow the **CS 328 HTML5 Coding Standards** as discussed in class and as given on the public course web site for all HTML5 documents.
- Remember to follow the **CS 328 CSS3 Coding Standards** as discussed in class and as given on the public course web site for all CSS3 that you write, also.

## Problem 1 - START THIS A.S.A.P. (in case there are PROBLEMS...)

You filled in, on your index card at the beginning of the semester:

- IF you had a database to use for additional application(s);
- its topic; and
- IF you were willing to share yours.

You might not remember at this point what you put! I have attempted to fill this in in the course Canvas gradebook (as a COMMENT in a 0-point assignment named **DB Status**); YOUR task, for this problem, is to do the following:

- Go to the course Canvas site.
- Click on the Grades link on the left-hand-side.

- Click on the assignment with the name "DB Status"; on the RIGHT-HAND-SIDE of that resulting window, you should see a COMMENT from me giving what your answers were on your index card (as far as I could tell)
  - if you DON'T? Please send me an e-mail with **Subject: CS 328 - NO DB Status COMMENT** including what, if anything, you CAN see. (You'll still need to finish this Problem by the homework deadline, so **CHECK THIS EARLY!!!**)
- **PART 1:** if you DO see this comment, ARE these still what you want? ARE these answers what you want as of NOW? You will send me a special e-mail to let me know.
  - NOTE that if you answered "Unsure" then -- you have to GIVE an answer now... 8-) And if your answer to Question 1 above was "No", then you'll likewise have a decision to make now.
  - IF (these answers are still correct) AND ( (none are UNSURE) OR (Answer1 is not NO) ):
    - send me an e-mail with **Subject: CS 328 - DB Status CONFIRMED**
    - and in this case, also make sure your "real" name is included in the body of your e-mail, also, because it isn't ALWAYS clear from your e-mail address.
    - [NOTE: if you answered OkToShare, this confirmation will let me know I can post versions of JUST your final model, final business rules, final db design, and final initial population on the course Canvas site for others to peruse and possibly use]
  - IF (any answers are NOT what you want now) OR (Answer1 was NO) OR (any were UNSURE):
    - send me an e-mail with **Subject: CS 328 - DB Status UPDATE**
    - and in this case, in the body, give your now-preferred-and-now-sure answers, also making sure your "real" name is included in the body of your e-mail, also.
    - If Answer 11 was NO or UNSURE, you must now decide on a database to use. As other class members confirm they are willing to share their databases, these will be posted to the course Canvas site. You will need to peruse these and decide on one, including its name as your revised Answer 2 (along with your Answer 1 that is now Yes!) (You can leave Answer 3 as NotApplicable...!)
    - [NOTE: if you answer OkToShare, this update will let me know I can post versions of JUST your final model, final business rules, final db design SQL script, and final initial population SQL script on the course Canvas site for others to peruse and possibly use]
  - ONCE you have sent this e-mail, you are DONE with **Part 1**. You must send this by the homework due date, but the EARLIER you send this, the BETTER (I can start posting "shared" database pieces on Canvas as soon as I start receiving permissions...)
- **Part 2:** AFTER sending the above required e-mail, you have some "pieces" to submit.
  - Submit the model as 325model.pdf, submit the business rules as 325biz-rules.pdf, submit the database design SQL script as 325design.sql, and submit the initial population SQL script as 325populate.sql for the database "base" you will be using (either yours, or those you have obtained from one of the shared databases on the course Canvas site)
  - also design an initial PHP document custom-login.php that allows someone to submit their username and password, and right now it JUST gives an appropriate welcome message to your database's scenario's applications.

This should use an initial external `custom.css` for your database scenario's applications, so you can maintain a consistent look-and-feel as you develop its applications.

- (You will likely be modifying these later -- they are just a starting point!)
- Submit these **six** files to complete **Part 2** of this problem.

## Problem 2

Create a SQL script `328hw7.sql`, and start it off with comments including your name, CS 328 - Homework 7, and the last-modified date.

Include a SQL\*Plus `spool` command to spool the results of running this SQL script to a file named `328hw7-out.txt`.

The purpose of this problem simply is to practice with a-queries-or-queries that you will be using more-general versions of in a later problem. I want to encourage you to remember that you can work with such queries on their own in SQL\*Plus, making sure they work as expected, before embedding them in code on one of the application tiers of an n-tier application.

Write a prompt or prompts that indicates that what is to follow is the current status for order number 11009

Now write a query or queries (your choice) that provide the following information: for order number 11009, what are:

- the date this order was placed
- the publisher for this order
- the book title(s) in this order
- the quantity received to-date for each book in this order

Follow this/these with a spool off command; the resulting files `328hw7.sql` and `328hw7-out.txt` should now be ready to submit.

## Problem 3

The purpose here is to practice using PHP to change a database (and explicitly commit the change), and to practice using bind variables within a SQL statement as well.

First: consider your external CSS3 file `bks.css` from the end of Homework 5. Make a **new** copy of this in a **different** directory, since you *might* be modifying it for this Homework 7 and you don't want to accidentally change how it styles Homework 5's files using it (since that could affect your Homework 5 grade!)

Now: consider the posted example `insert-dept.php` from Week 8 - Lecture 1. Surely a bookstore could use a way to enter information about a new publisher from which they would like to purchase new titles.

Create a PHP document `insert-pub.php`, styled using `bks.css`, which meets the following specifications:

- This document should be one of those that can generate an initial form or that form's action, depending on the keys that exist in the `$_POST` array when it is reached.
- Each page that this document generates should include, in its body element, your name, CS 328, and the name of your bookstore from `bks-splash.html`.

- Its initial form should require that the user enter an Oracle username, password, and information on a new publisher.
- The action for that form, when submitted, is to insert an appropriate row into the `publisher` table and commit the change.
  - Make sure that you use bind variables appropriately to help thwart SQL injection -- and you can assume that none of the publisher data should legitimately contain HTML tags, and so you can strip any that someone attempts to include.
- Make sure the response page shown to the user summarizes what has been inserted.
- The response page should also include a hypertext link with appropriate text that links back to `insert-pub.php`.
  - (my understanding is that a link such as this should be treated like a request whose `method="get"`... so following this link should cause what to be displayed? That's a rhetorical question to think about, NOT one you have to answer for this problem... 8-) )

#### An **OPTIONAL** VARIATION:

- Some users might prefer to let the system determine and set the publisher numbers; if you would like to have PHP determine (with help from the DBMS) what publisher number should be used for the new publisher, you may do so (note that, in this case, the user WON'T be asked to enter a publisher number).

Do **NOT** add a link to your `insert-pub.php` from your `index.html` on `nrs-projects`; everyone is practicing pretty much the same thing here. DO include the URL one can use to run your `insert-pub.php` on `nrs-projects` in its opening comment.

Submit your `insert-pub.php`; you will be submitting Homework 7's `bks.css` at the end of Homework 7 (but of course make sure it still works on these documents, also! And, it is always OK to turn in versions-in-progress along the way!).

## Problem 4

You can use bind variables in a SQL `SELECT` statement, also.

First: Fun fact #1: You can use PHP's `sprintf` function to get a string version of a number formatted as desired (for example, to a set number of fractional places).

- You can read more about it in the PHP manual, but, for example, this statement sets a variable `$formatted_price` to a string containing a string depiction of formatting a number `$price` to 2 fractional places:

```
$formatted_price = sprintf("$%.2f", $price);
```

Second: Fun Fact #2: A bind variable CAN be used for right-hand-side of a `LIKE` operation in a SQL `SELECT` statement's `WHERE` clause -- but the value bound TO that bind variable has to include any desired `%` or `_` wildcard characters. (Remember: `%` match any 0-or-more characters, `_` matches any single character.)

**ASSUME that employees of your bookstore can receive a 15% discount off a title's retail price.** Create a PHP document `empl-disct-query.php`, styled using `bks.css`, which meets the following specifications:

- This document should be one of those that can generate an initial form or that form's action, depending on the keys that exist in the `$_POST` array when it is reached.

- Each page that this document generates should include, in its body element, your name, CS 328, and the name of your bookstore from `bks-splash.html`.
- This page should use your Homework 7 copy/version of `bks.css` (and you may add to it if you wish, but make sure the resulting version works with all of Homework 7's bookstore-related PHP documents).
- Its initial form should require that the user enter an Oracle username, password, and a desired `title_name` search string (which CAN contain `%` or `_` wildcard characters)
- The action for that form, when submitted, is to project the following:
  - `title_name`
  - `current_qty_on_hand` (simply to let the employee know if there are enough on hand for them to purchase)
  - the employee discount price for that title (the `title_price` with the 15% employee discount applied)

...but ONLY for titles whose `title_name` match the employee-entered search string when the `LIKE` operation is involved.
- Further requirements for the handling-a-submitted-form part:
  - You are **REQUIRED** to use a bind variable for the desired `title_name` search string. While you should permit any `%` or `_` characters in the entered search string, you can assume that none of these strings should legitimately contain HTML tags, and so you can strip out any that someone attempts to include.
  - Remember to use PHP's `sprintf` function to ensure that you print the employee-discount price(s) to exactly 2 fractional places
  - There **COULD** be **more than one** row in response to this -- there also could be **zero** rows (no matches)

You are expected to use a `table` element for displaying the query results in an attractive, readable form; it is OK if you end up giving a table header row and no table content rows for no-rows-found.
- The response page should also include a hypertext link with appropriate text that links back to `empl-disct-query.php`.

Do **NOT** add a link to your `empl-disct-query.php` from your `index.html` on `nrs-projects`; everyone is practicing pretty much the same thing here. DO include the URL one can use to run your `empl-disct-query.php` on `nrs-projects` in its opening comment.

Submit your `empl-disct-query.php`; you will be submitting Homework 7's `bks.css` at the end of Homework 7 (but of course make sure it still works on these documents, also! And, it is always OK to turn in versions-in-progress along the way!).

## Problem 5

Now: consider again the query-or-queries you wrote for Problem 1. What if you would like this information for *any* order number entered (rather than always for order 11009)? What might an HTML5 form look like that could submit a desired order number to the application tier, so that something on the application tier could build the appropriate query/queries and send the resulting query/queries to the data tier?

Create a PHP document `get-order-status.php`, styled using `bks.css`, which meets the following specifications:

- This document should be one of those that can generate an initial form or that form's action, depending on the keys that exist in the `$_POST` array when it is reached.
- Each page that this document generates should include, in its body element, your name, CS 328, and the name of your bookstore from `bks-splash.html`.
- This page should use your Homework 7 copy/version of `bks.css` (and you may add to it if you wish, but make sure the resulting version works with all of Homework 7's bookstore-related PHP documents).
- Its initial form should require that the user enter an Oracle username, password, and an order number of interest.
  - for THIS Homework, use a **textfield** for entering the order number --
  - (for a future homework [after we have covered sessions], I hope to remember to have you refactor this to use a dynamically-populated drop-down box instead...!)
- The action for that form, when submitted, is to use the form-provided information to query the database for:
  - the date the specified order was placed
  - the publisher for this order
  - the book title(s) in this order
  - the quantity received to-date for each book in this order
- Further requirements for the handling-a-submitted-form part:
  - Make sure that you use a bind variable appropriately for the desired order number -- and you can assume that none of the order numbers should legitimately contain HTML tags, and so you can strip any that someone attempts to include.
  - There COULD be **more than one** row in response to this (multiple titles involved in a single order) -- there also could be **zero** rows (for a non-existent order number).  
 You are expected to use a `table` element for displaying at least the book title(s) and their quantity received to-date for each; I'll leave it up to you whether you'd like to include the order data and publisher within this `table` element, or separately.  
 Either way, display these results in an attractive, readable form; it is OK if you end up giving a table header row and no table content rows in the no-rows-found case.
- The response page should also include a hypertext link with appropriate text that links back to `get-order-status.php`.

Do **NOT** add a link to your `get-order-status.php` from your `index.html` on `nrs-projects`; everyone is practicing pretty much the same thing here. DO include the URL one can use to run your `get-order-status.php` on `nrs-projects` in its opening comment.

Submit your `get-order-status.php`; you will be submitting Homework 7's `bks.css` at the end of Homework 7 (but of course make sure it still works on these documents, also! And, it is always OK to turn in versions-in-progress along the way!).

## Problem 6

The purpose here is to practice calling a PL/SQL stored procedure from PHP.

Consider Homework 4, Problem 1's PL/SQL stored procedure `insert_order_needed`, whose parameters are the desired ISBN and order quantity for the desired order.

We know there are times that `insert_order_needed` is called automatically (when the stock of a given title goes below a certain point) -- BUT savvy bookstore managers may also hand-enter such rows, based on their advanced local market knowledge (an upcoming film festival, for example, or a "hot" new trend).

SO -- Create a PHP document `hand-insert-order-needed.php`, styled using `bks.css`, which meets the following specifications:

- This document should be one of those that can generate an initial form or that form's action, depending on the keys that exist in the `$_POST` array when it is reached.
- Each page that this document generates should include, in its body element, your name, CS 328, and the name of your bookstore from `bks-splash.html`.
- This page should use your Homework 7 copy/version of `bks.css` (and you may add to it if you wish, but make sure the resulting version works with all of Homework 7's bookstore-related PHP documents).
- Its initial form should require that the user enter an Oracle username, password, the desired ISBN, and the desired order quantity for the suggested order-needed.
  - for THIS Homework, use a **textfield** for entering the ISBN --
  - (for a future homework [after we have covered sessions], I hope to remember to have you refactor this to use a dynamically-populated drop-down box instead...!)
- Further requirements for the handling-a-submitted-form part:
  - You are required to call Homework 4, Problem 1's PL/SQL stored procedure `insert_order_needed` (an example version of which is available on the course Moodle site under "Selected solutions" if you have any qualms about your version)
  - Make sure that you use bind variables appropriately for the desired ISBN and desired quantity in your stored procedure call --  
and since neither of these should legitimately contain HTML tags, you can reasonably strip any that someone attempts to include.
  - Make sure you commit the result!
  - Also display a descriptive message describing the action just completed in a readable/reasonably attractive way.
- The response page should also include a hypertext link with appropriate text that links back to `hand-insert-order-needed.php`.

Do **NOT** add a link to your `hand-insert-order-needed.php` from your `index.html` on `nrs-projects`; everyone is practicing pretty much the same thing here. DO include the URL one can use to run your `hand-insert-order-needed.php` on `nrs-projects` in its opening comment.

Submit your `hand-insert-order-needed.php`; you will be submitting Homework 7's `bks.css` at the end of Homework 7 (but of course make sure it still works on these documents, also! And, it is always OK to turn in versions-in-progress along the way!).

## Problem 7

The purpose here is to practice calling a PL/SQL stored function from PHP. (OK, **two** stored functions... 8-) )

Consider Homework 3 - Problem 2's PL/SQL stored function `is_on_order`, and Homework 4 - Problem 2's PL/SQL stored function `pending_order_needed`.

Both of these happen to expect an ISBN as their single parameter -- and then `is_on_order` returns boolean `true` if that ISBN is currently actually on-order, while `pending_order_needed` returns boolean `true` if that ISBN currently has a pending, not-yet-handled `order_needed` row.

We've seen how these can be used as part of other PL/SQL stored subroutines -- and hopefully you can see that a savvy bookstore manager might like to ask these questions conveniently as well (before, perhaps, deciding to hand-add a new `order_needed` row... 8-) ).

SO -- Create a PHP document `current-isbn-status.php`, styled using `bks.css`, which meets the following specifications:

- This document should be one of those that can generate an initial form or that form's action, depending on the keys that exist in the `$_POST` array when it is reached.
- Each page that this document generates should include, in its body element, your name, CS 328, and the name of your bookstore from `bks-splash.html`.
- This page should use your Homework 7 copy/version of `bks.css` (and you may add to it if you wish, but make sure the resulting version works with all of Homework 7's bookstore-related PHP documents).
- Its initial form should require that the user enter an Oracle username, password, and the desired ISBN;
  - for THIS Homework, use a **textfield** for entering the ISBN --
  - (for a future homework [after we have covered sessions], I hope to remember to have you refactor this to use a dynamically-populated drop-down box instead...! Indeed, something somehow combining Problems 4 and 5 might be interesting...? Hmmmm.)
- Further requirements for the handling-a-submitted-form part:
  - You are required to call Homework 3 - Problem 2's `is_on_order`, and Homework 4 - Problem 2's `pending_order_needed` (example versions of which is available on the course Moodle site under "Selected solutions" if you have any qualms about your versions)
  - Make sure that you use bind variables appropriately for the desired ISBN in each of your stored function calls --

and since this should not legitimately contain HTML tags, you can reasonably strip any that someone attempts to include.

  - Include descriptive output indicating, based on what the stored functions return, whether or not this ISBN is currently on order or not, **and** whether or not this ISBN currently has any pending, not-yet-acted-upon `order_needed` rows.
- The response page should also include a hypertext link with appropriate text that links back to `current-isbn-status.php`.

Do **NOT** add a link to your `current-isbn-status.php` from your `index.html` on `nrs-projects`; everyone is practicing pretty much the same thing here. DO include the URL one can use to run your `current-isbn-status.php` on `nrs-projects` in its opening comment.



Submit your `current-isbn-status.php`; and your Homework 7's `bks.css` should be complete when it nicely formats this PHP document as well as those from the previous problems! (And, it is always OK to turn in versions-in-progress along the way!)