CS 328 - Homework 6

Deadline

11:59 pm on **Friday, March 13, 2026**

Purpose

To look over the class CSS style standards, to keep in practice with PL/SQL by writing a small stored function, to think about color and accessibility, and to practice more with CSS external style sheets.

How to submit

You complete **Problem 1** on the course Canvas site (short-answer questions on CS 328 CSS coding standards), so that you can see if you are on the right track.

Each time you wish to submit your work-so-far for **Problems 2 onward**, submit your files using **`~st10/328submit`** on nrs-projects, with a homework number of **`6`**.

Important note: It is quite likely that your PL/SQL files will be in a different directory than your HTML files. That's fine, and preferable!

- Just remember that you need to run **`~st10/328submit`** from **EACH** directory with files to be submitted for Homework 6.

More important notes

- Remember: You are required to use **`normalize.css`** for all of your web pages for CS 328, and to add **`link`** elements for additional CSS external stylesheets *after* this (but still within the **`head`** element).

  EXCEPT for **`normalize.css`**, **DO NOT USE ANY CSS FRAMEWORKS or PREDEFINED LIBRARIES for this course** (unless you get prior, explicit permission). One of this course's purposes is to provide you with some practice with the basics of "plain" CSS, so you can better make use of frameworks and predefined libraries later.

- Remember that there are **CS 328 CSS Coding Standards so far** posted on the public course web site, under **References** -- you are also expected to follow these for all course documents.
  - External style sheets are expected to validate as valid CSS using the validator link included in the **CS 328 CSS Coding Standards so far**.
- You are expected to follow all course coding standards posted on the public course web site; course documents are also expected to validate as strict-style HTML.

Problem 1 - 14 points

Purpose: to review the CS 328 class style and coding standards for CSS

Task: Answer the "HW 6 - Problem 1 - Short-answer questions on CS 328 CSS coding standards" on the course Canvas site.

Criteria: You will receive full credit for giving the correct answer for each question.

Problem 2 - PL/SQL stored function `get_order_date`

Purpose: to get more practice writing a PL/SQL stored function, and to provide more ideas for a future PL/SQL stored function for your "second" database

Problem 2 background

Consider the bookstore database's **`order_stock`** table, representing orders of stock (books) from a publisher for this bookstore:

```
create table order_stock
(ord_num           number(6) generated by default as identity,
 pub_id            number(3) not null,
 ord_date_placed   date      not null,
 ord_date_complete date,
 constraint        order_stock_pk     primary key (ord_num),
 constraint        order_stock_fk_pub foreign key (pub_id)
                   references publisher
);
```

Notice that **`order_stock`**'s **`ord_date_placed`** attribute is specified as **`not null`** -- every order of stock must have a date that it was placed.

Also consider the useful function **`to_char`** that allows you to specify a date format in a variety of attractive customized ways. (You can review some of its format string options in the posting "**Some string- and date- and time-related SQL functions**" on the CS 328 **PL/SQL References** page.)

Assume that this bookstore has several different reports in which bookstore workers would prefer that order dates be in the format:

'<3-letter-day of the week>, <3-letter-month> <two-digit day of the month>, <4-digit year>'

For example: using this format, this homework's deadline date would be:

```
'Fri, Mar 13, 2026'
```

A small function that returns an order's date-placed in this format might be useful.

2 part a - start setting up your SQL script

Task: Start a SQL script **`328hw6.sql`** as follows:

- Create this file, and protect it:

  `chmod 600 328hw6.sql`

- Include:
  - comments containing at least your name, **`CS 328 - Homework 6`**, and the last-modified date
  - followed by a SQL*Plus **`spool`** command to spool the results of running this SQL script to a file named **`328hw6-out.txt`**

  - followed by a **`prompt`** command including your name
- **Be sure** to **`spool off`** at the **end** of this script (after creating and testing your stored function below).

2 part b - create stored function `get_order_date`

Task: In your script **`328hw6.sql`**, create and compile a PL/SQL stored function **`get_order_date`** that meets the following requirements:

- It expects one parameter, an order number for an order of stock.
- It returns a string version of the date that order was placed, formatted as described above, as `'`<3-letter-day of the week>`,` <3-letter-month> <two-digit day of the month>`,` <4-digit year>`'`
  - However, if given a non-existent order number. it should return the string `'Not an order number'`.
- Create an opening comment block for your function that has a **`function:`** part and **`purpose:`** part in the same style as used in posted class examples. (You don't have to give an `examples:` part, but you can if you wish.)
- Follow that with the PL/SQL code creating your function.
- Remember to follow your PL/SQL function with:

  `/`

  `show errors`
- Then put a comment saying you are about to **test** your function **`get_order_date`**.
- Follow that with at least **THREE** tests of **`get_order_date`**:
  - one for the order number `11009`
  - one for the order number `11014`
  - one with a **non-existent** order number (one NOT in the `order_stock` table)

  ...**EACH** test including:
  - **`prompt`** command(s) stating that you are about to test **`get_order_date`** and **describe** what you should see if it is working properly.
    - (Your description should be specific enough that someone looking just at the spooled output can tell if the test passed or not.)
  - Remember that:
    - You will need to declare a SQL*Plus local variable to hold the result returned by your function.
    - The **`exec`** command is a little **different** when calling a function than when calling a procedure.
    - You can use the **`print`** command to display the value of a SQL*Plus local variable.

ALTERNATIVE option for TESTING `get_order_date`:

- You may use Homework 4 - Problem 7's PL/SQL stored procedure **`print_test`** for your tests,

  first including a **`prompt`** command to state that you are testing **`get_order_date`**, and calling **`set serveroutput on`** since `print_test` uses `dbms_output.put_line`,

  and then, for each of your tests:

  - using a first argument to **`print_test`** that is a **string** containing a boolean expression containing an example call to **`get_order_date`** and what it should return,
  - and using a second argument to **`print_test`** that is that boolean expression.

- For example,

  (recalling that you print a single quote in SQL by putting two single quotes,

  and that when you want to **extend** a SQL*Plus command to a **next** line, you must end the **first** line with a **DASH**):

```
exec print_test('get_order_date(11009) = ''Wed, Dec 10, 2025''',-
                get_order_date(11009) = 'Wed, Dec 10, 2025')
```

  ...(noting that, because of how `sysdate` is used in `pop-bks.sql`, `11009`'s `ord_date_placed` value is probably DIFFERENT for your version of the bookstore database...!)

Make sure you have a **`spool off`** command at the end of **`328hw6.sql`**, and submit your files **`328hw6.sql`** and **`328hw6-out.txt`**.

Problem 3 - Colors and Accessibility

Purpose: to learn more about some aspects of accessibility related to CSS color choices, and to practice selecting and testing color and background color choices.

Problem 3 background:

We have discussed how you can use CSS to specify the color and background color of HTML elements. Also, we have said that accessibility is an important goal for HTML documents.

If a person has color vision deficiency, or is trying to read something in bright conditions, or is using a screen happening to be lacking in terms of size, resolution, and/or contrast, an HTML document being displayed with too-little color contrast between text and background may be difficult, or even impossible, to read. (reference: "Colour contrast - why does it matter?", at https://accessibility.blog.gov.uk/2016/06/17/colour-contrast-why-does-it-matter/)

In addition to the Color Picker in Section 3.4 of the zyBooks course text, there are many color tools on the Web -- for example:

- Sessions College Color calculator, at https://www.sessions.edu/color-calculator/
- W3Schools intro to HSL and HSLA's HSL Calculator at https://www.w3schools.com/colors/colors_hsl.asp

There are numerous color contrast tools as well. Here is one example of a tool, snook.ca's Colour Contrast Check, that lets you specify a foreground and background color, and then helps you determine

if they provide enough contrast:

- snook.ca's Colour Contrast Check tool, at https://snook.ca/technical/colour_contrast/colour.html

Such tools can be useful for testing a foreground and background color you are interested in using, to see if they provide sufficient contrast. But it can be tough to use this to determine pairs of such colors to begin with.

Happily, there are numerous tools that suggest accessible possibilities -- for example:

- Accessible Colour Contrast, at https://www.sussex.ac.uk/tel/resource/tel_website/accessiblecontrast/

...can be used to help determine accessible combinations that "...meet WCAG 2.1 Level AA or higher for the contrast between text and background".

your task for Problem 3:

- Make a file **`328hw6-color.txt`**, and start this file with your **name** and the **last modified date**.
- Determine a foreground color and background color pair -- **at least one** of which is **not** black (`#ffffff`) or white (`#000000`) -- you are interested in using later in this homework that is **at least WCAG 2 AA Compliant** according to the tester at https://snook.ca/technical/colour_contrast/colour.html
  - (That is, make sure it shows **YES** in the **fifth** textfield in the **Results** section for your chosen pair of colors.)
- Demonstrate that you did so by turning in the following **for at least one WCAG 2 AA Compliant** pair of colors:
  - include your **foreground** color choice, given in hexadecimal format (e.g., `#046b99`)
  - include your **background** color choice, given in hexadecimal format
  - include the **brightness difference** for these (from the **first** textfield in the **Results** section)
  - the **color difference** for these (from the **second** textfield in the **Results** section), and
  - the **contrast ratio** for these (from the **fourth** textfield in the **Results** section).

  ... in https://snook.ca/technical/colour_contrast/colour.html.

Submit your file **`328hw6-color.txt`**.

Problem 4 - adding style to your `about-bks.html`

Purpose: to get more practice styling a document using external CSS.

Task: Consider your HTML document **`about-bks.html`** from Homework 2 - Problem 3, describing your chosen theme for your instance of the bookstore database.

Make a **COPY** of your **`about-bks.html`** in a **DIFFERENT** directory on nrs-projects (so you will not interfere with your earlier homework's files!).

Design an initial version of an external style sheet **`bks.css`** for this homework's copy of **`about-bks.html`** (and hopefully/possibly for other future bookstore-related documents). Modify *this* homework's copy of **`about-bks.html`** to use **both `normalize.css`** and this new style sheet

`bks.css`.

Requirements for this initial version of **`bks.css`**:

- Include a comment including at least your **name** and the last-modified date.
- Include a rule styling the **`body`** element, that includes at least a declaration noticeably styling its **`margin`** or **`margin-left`** property.
- Include at least **five** additional, distinct rules that have a visible effect (but you can certainly add more if you are inspired!). (So, your `bks.css` should have at least six rules.)
- Have at least **one** rule visibly specifying the **`color`** or **`background-color`** property of some element(s).
  - Make sure the text/background contrast for your styled document is **at least WCAG 2 AA Compliant** based on the tester at https://snook.ca/technical/colour_contrast/colour.html.
- As discussed in the zyBooks course text in Chapter 3, Section 3.5, "Font and text properties", there are **five font-related properties** in addition to the shortcut property **`font`**.
  - For **this** problem, for experimenting purposes, **do not use** the shortcut property **`font`** -- instead, use **at least three** of the **non-shortcut** font-related properties amongst your rules, such that the results are clearly visible in your styled document.
  - Note: remember the follow the course style standards with regard to specifying sizes!
- Add additional rules as desired to further attractively format elements of this document.

Make sure your resulting documents successfully validate as strict-style HTML and valid CSS. (If you would find it convenient, you can add a link to the W3C CSS validator at https://jigsaw.w3.org/css-validator/ to your document's `footer` element.)

Submit this homework's modified copies of **`about-bks.html`** and **`about-bks.xhtml`** along with your **`bks.css`**.

Problem 5 - formatting a `table` element

Purpose: to practice styling a **`table`** element (especially since its default style is often not what you want)

Task: Starting from the CS 328 **`html-template.html`**, create a strict-style HTML document whose name includes **`prob5`**, styled by an external CSS style sheet whose name also includes **`prob5`**.

Your **`*prob5*.html`** should meet the class style standards as well as the following requirements:

- Fill in the opening comment block as specified, putting in your **name**, the last modified **date**, and the **URL** that can be used to run your document.
  - (You *will* lose some credit if this URL does not work when I or the grader paste it into a browser.)
- Within its **`head`** element, give the **`title`** element appropriate descriptive content.
- Within its **`head`** element, *after* the **`link`** element styling this document with **`normalize.css`**, add a second **`link`** element so that this will be *further* styled using an external CSS style sheet **`*prob5*.css`**.

Within your **`*prob5*.html`**'s **`body`** element:

- Include an appropriate **`h1`** element.
- Include a **`table`** element, whose subject(s) is/are your choice, that includes at least the following:
  - at least **5** rows and **2** columns
  - an appropriate **`caption`** element
  - appropriate **`th`** elements, with appropriate **`scope`** attributes
  - (It is fine to use a copy of your Homework 3 - Problem 2 table if you wish -- it should meet the above requirements.)
- Within its **`footer`** element, add a **`p`** element whose content includes **your name**.

Your **`*prob5*.css`** should meet the following requirements:

- Include a comment including at least your **file's** name, *your* **name**, and the last-modified date.
- Include a rule styling the **`body`** element, that includes at least a declaration noticeably styling its **`margin`** or **`margin-left`** property.
- Use foreground and background colors that are **at least WCAG 2 AA Compliant** based on the tester at: https://snook.ca/technical/colour_contrast/colour.html
- Include at least one rule adding an attractive **border** to the **`table`**, **`td`**, and **`th`** elements.
  - You also should make appropriate use of the **`border-collapse`** property to prevent a doubled-border in the resulting displayed table.
- At least one rule should affect one or more of the table-related elements' **`padding`** values in a visible and attractive way.
- At least one rule should affect one or more of your document's elements' **font** values in a visible and attractive way.
  - For this problem, it is up to you whether you use the shortcut `font` property or individual font-related properties.
  - Note: remember the follow the course style standards with regard to specifying sizes!
- Add additional rules as desired to further attractively format elements of this document.

Make sure your resulting documents successfully validate as strict-style HTML and valid CSS. (If you would find it convenient, you can add a link to the W3C CSS validator at https://jigsaw.w3.org/css-validator/ to your document's `footer` element.)

Submit your **`*prob5*.html`**, **`*prob5*.xhtml`**, and **`*prob5*.css`** files.