

Applying SQL #2

(25 points • est. time 75 minutes)

Background

This exercise has you use material you've seen up through chapter 3 of our Oracle SQL text and the material on formatting output in SQL*Plus. You will use objects in the Issue25 schema. Use the schema handout as a reference.

Tasks

1. Launch *SQL*Plus* and connect to our *cis119do* database.
2. Spool the results of your session to a file.
3. Issue the following command:

```
SQL> SELECT user, TO_CHAR(sysdate, 'MON-DD HH:MI am') from DUAL;
```
4. Show the course number, description, and cost for courses that have no prerequisite and whose course number is less than 100. Sort the results in descending order of cost.
5. Show the last name, first name, and contact code for each writer. When no contact is available, display the text **No contact available** instead. Have the column heading for the contact values say **CONTACT#**.
6. Show the complete name (eg: Lawton, Pat) and amount for only the freelancers. Also include a column of calculated results that shows what the amount would be if each freelancer were given a 27.3% raise, with the calculated amount *always rounded up* to the next whole dollar.
7. Show the title and type of any BUS or ADV or FMK article. Use the DECODE function to display the text Business, Advertising, or Financial Market instead of the 3-character type abbreviation. The output should look like:

Title	Article Type
-----	-----
Trans-Alaskan Oil Pipeline Opening	Business
AT&T Antitrust Settlement	Business
Building Trade Outlook	Business
Cola Advertising War	Advertising
Stock Market's Black Monday	Financial Market
...	

8. Show the title, type, issue and length for articles whose title is longer than 30 characters and whose length is less than 2000 words.
9. Display the description and cost for any course having **Unix** anywhere in its title. Use *string function(s)* to locate the articles. Do *not* use wildcard(s).
10. A particular cruise ship has 1765 total people aboard. If each lifeboat can hold 88 people, how many lifeboats are needed to ensure there are enough for everyone?
11. Use string functions to display a *nonduplicated* list of student salutations, with periods removed (eg: Mr. should display as Mr). Do **not** use DECODE.

12. Show each article's title and a descriptive word indicating the article's length. For articles up to 1,000 words in length, display the word **short**; for those longer than 1,000 and up to 2,500 words, display the word **medium**; for those longer than 2,500 and up to 4,000 words, display the word **long**. Any article longer than 4,000 words should be labeled **novel**. Your output should resemble the following. Note that the rows are sorted by title.

TITLE	LENGTH
-----	-----
\$100 Billion AOL Time Warner Merger Approved	Medium
25% Tax Cut Bill Approved	Medium
AT&T Antitrust Settlement	Medium
Advertising Over the Past 25 Years	Long
Alternative Energy Sources	Medium
Bingham Family Feud	Medium
Building Trade Outlook	Medium
Bush Finally Wins!	Novel
...	

13. My schema includes a table named `pledge` that stores data for a bowling fundraiser in which people pledge either a flat amount (such as \$5.00), an amount per pin knocked down (eg: \$0.50), or a combination of both a flat amount and a per pin amount. List its structure (`descr ttrollen.pledge`) and then develop a query that calculates **TOTAL PLEDGE**, as illustrated below.

PLEDGEID	SCORE	FLAT_AMOUNT	PER_PIN_AMOUNT	TOTAL PLEDGE
-----	-----	-----	-----	-----
1	100	50		50
2	200	5	1	205
3	90		.5	45
4	171		2	342
5	255	10	1	265
6	134		.1	13.4
7	100	5	5	505

14. Turn off spooling.
15. Open your spooled text file in a word processor. Change the font to `Courier New` and use a font size of 8 or 10.
16. Edit the file to *remove* mistakes. Do not type any new commands or simulated output.
17. Type the task number (3-13) to identify each successful command.
18. Print your spooled text file.