CS 328 - Week 5 Lab Exercise - 2026-02-19

Deadline

Due by **11:59 pm on Friday, February 20**

Purpose

To practice more with PL/SQL, including writing a stored function and a stored procedure, and including practice with parameters, exception-handling, and a cursor-controlled **`for`** loop.

How to submit

Submit your files for this lab using **`~st10/328submit`** on nrs-projects, each time entering a lab number of **85**.

Requirements

- For **THIS particular** lab exercise, you may work individually OR use pair programming, **your choice**!
  - If you do use pair programming, make sure BOTH of your names appear in each file submitted!
- Whether working individually or using pair programming, make sure you have a copy of your lab exercise files, and that you submit them using `~st10/328submit` on nrs-projects, with a lab number of **85**.
- You are expected to follow the style standards from the posted "CS 328 SQL and PL/SQL Coding Standards so far" (at https://nrs-projects.humboldt.edu/~st10/s25cs328/328-sql-plsql-coding-standards.pdf).

Lab set-up

- On nrs-projects, **CREATE** a new directory **`328lab5`**, **protect** it, and go to it:

```
mkdir 328lab5
chmod 700 328lab5
cd 328lab5
```

- If the driver has not previously executed **`set-up-ex-tbls.sql`** in their Oracle account, they should do so, so that they have the tables `empl`, `dept`, and `customer` in their database.
  - If needed, they can get a copy of this script using:

```
cp ~st10/set-up-ex-tbls.sql .    # don't forget the blank and period!
```

- Start your initial version of a SQL script **`lab5.sql`** by making a copy of the provided file **`lab5-START.html`**:

```
cp  ~st10/lab5-START.sql  lab5.sql
```

  - In the interests of time, this provides an **opening comment block** and **testing code** for the PL/SQL subroutines you will be writing and modifying for this lab exercise.
- In your file **`lab5.sql`**:

- Replace **ALL THREE** instances of **`PUT YOUR NAME(S) HERE`** (one in the opening comment, one in a **prompt** command, and one of Problem 2's opening comment) with both of your first *and* last names.

Problem 1 - stored function `num_pd_more`

To get some practice writing a PL/SQL stored function, in your SQL script **`lab5.sql`**, write a stored function **`num_pd_more`** that meets the following requirements:

- Write it **after** the provided opening comment block for this function,

  but **before** its provided **`/`** and **`show errors`** and testing code.

- It expects a lower-limit salary value.
  - Note: use the type **`number`** for this parameter.
- It returns the number of employees in the **`empl`** table whose salary is strictly greater than that given lower-limit salary value.

If successful, your resulting **`lab5-out.txt`** should show that your function successfully compiled, and that its tests passed.

Problem 2 - some light PL/SQL exception-handling practice

Consider: in the `empl` table created in **`set-up-ex-tbls.sql`**, the **`mgr`** attribute is a foreign key referencing **`empl`**'s **`empl_num`** attribute -- **`mgr`** is the employee number of that employee's manager.

One attempt at a PL/SQL stored function **`get_manager`**, that expects an employee's last name and returns the last name of that employee's manager, already has been included in the **`lab5.sql`** you copied from **`lab5-START.sql`**.

But, when you run this, it has some definite issues, that you should be able to see when you run this in `sqlplus /`.

So: **MODIFY** this function **`get_manager`** so that it meets the following requirements:

- Insert *your* **names** in the comment that specifies that you do so
- Add **EXCEPTION HANDLING** to the function **`get_manager`** to handle the tests that currently do not pass, so that they now DO pass.

If successful, your resulting **`lab5-out.txt`** should show that your modified version of **`get_manager`** successfully compiled, and that its tests passed.

Problem 3 - stored procedure `list_managers`

Now that you have **`get_manager`**, that allows for a good opportunity to practice calling a PL/SQL function from another PL/SQL subroutine, and also allow for a good excuse to practice with a cursor-controlled **`for`** loop.

In your SQL script **`lab5.sql`**, write a PL/SQL procedure **`list_managers`** that meets the following requirements:

- Write it **after** the provided opening comment block for this procedure,

  but **before** its provided **`/`** and **`show errors`** and testing code.

- It expects a job title.
- It prints to the screen, for each employee with that job title:
  - their last name,
  - then a blank and a dash and a blank,
  - then **`managed by:`** followed by the last name of their manager.
- (And it returns nothing, since it is a procedure!)
- If there are no employees with that job title, it should simply print a message to the screen that includes the nonexistent job title, and notes that there are no employees with that job title.
- For full credit:
  - it is required to use an **if** statement.
  - it is required to make appropriate use of a cursor-controlled **`for`** loop.
  - it is required to appropriately call Problem 2's function **`get_manager`** to get the last names of the managers for the employees with that job title.

If successful, your resulting **`lab5-out.txt`** should show that your procedure successfully compiled, and that its tests passed.

Make sure that you submit the files:

- **`lab5.sql`** and **`lab5-out.txt`**

...using **`~st10/328submit`** on nrs-projects, with a lab number of **85**.

(if you used pair-programming):
How the navigator can get files `lab5.sql` and `lab5-out.txt`:

(for a driver with username *dr12*, and a navigator with username *na89* - replace these with your *actual* usernames when you actually do this)

- The **driver** should ***temporarily*** make their directory **`328lab5`** with these files world-executable, and these files world-readable:

```
chmod 711 .    # notice the space and the period!
chmod 644 lab5.sql lab5-out.txt
```

- Now the **navigator** can copy these into a directory of their choice.

  For example, assuming that the navigator created their **`328lab5`** in their home directory, and that the navigator has **`cd`**'d to the directory they want to copy the driver's files into:

```
cp ~dr12/328lab5/lab5.sql .   # notice the space and the period!
cp ~dr12/328lab5/lab5-out.txt .
```

- After the copies are successfully made, the **driver** and **navigator** should **BOTH protect** these files:

```
chmod 700 .    # notice the space and the period!
chmod 600 lab5.sql lab5-out.txt
```

- ...and **both** can now submit these using **`~st10/328submit`** from their directory containing these files.