Exercises for Chapter 20: Functions

Try It Yourself

The projects in this section are meant to have you utilize all of the skills that you have acquired throughout this chapter. Here are some exercises that will help you test the depth of your understanding.

1) Write a stored function called `new_student_id` that takes in no parameters and returns a `student.student_id%TYPE`. The value returned will be used when inserting a new student into the CTA application. It will be derived by using the formula: `student_id_seq.NEXTVAL`.

Answer: The function should look similar to the following:

```
CREATE OR REPLACE FUNCTION new_student_id
RETURN student.student_id%TYPE
AS
   v_student_id student.student_id%TYPE;
BEGIN
   SELECT student_id_seq.NEXTVAL
     INTO v_student_id
     FROM dual;
   RETURN(v_student_id);
END;
```

---

2) Write a stored function called `zip_does_not_exist` that takes in a `zipcode.zip%TYPE` and returns a Boolean. The function will return TRUE if the zipcode passed into it does not exist. It will return a FALSE if the zipcode exists. Hint: An example of how it might be used is as follows:

For Example

```
DECLARE
   cons_zip CONSTANT zipcode.zip%TYPE := '&sv_zipcode';
   e_zipcode_is_not_valid EXCEPTION;
BEGIN
   IF zipcode_does_not_exist(cons_zip) THEN
      RAISE e_zipcode_is_not_valid;
   ELSE
      -- An insert of an instructor's record that
      -- uses of the checked value of zipcode might go here.
```

```
      NULL;
   END IF;
EXCEPTION
   WHEN e_zipcode_is_not_valid THEN
      RAISE_APPLICATION_ERROR
         (-20003, 'Could not find zipcode '||cons_zip||'.');
END;
```

Answer: The function should look similar to the following:

```
CREATE OR REPLACE FUNCTION zipcode_does_not_exist
   (i_zipcode IN zipcode.zip%TYPE)
RETURN BOOLEAN
AS
   v_dummy char(1);
BEGIN
   SELECT NULL
     INTO v_dummy
     FROM zipcode
    WHERE zip = i_zipcode;

   -- meaning the zipcode does exits
   RETURN FALSE;
EXCEPTION
   WHEN OTHERS THEN
      -- the select statement above will cause an exception
      -- to be raised if the zipcode is not in the database.
      RETURN TRUE;
END zipcode_does_not_exist;
```

---

3) Create a new function. For a given instructor, determine how many sections he or she is teaching. If the number is greater than or equal to 3, return a message saying the instructor needs a vacation. Otherwise, return a message saying how many sections this instructor is teaching.

Answer: The function should look similar to the following:

```
CREATE OR REPLACE FUNCTION instructor_status
   (i_first_name IN instructor.first_name%TYPE,
    i_last_name  IN instructor.last_name%TYPE)
RETURN VARCHAR2
AS
   v_instructor_id instructor.instructor_id%TYPE;
   v_section_count NUMBER;
   v_status        VARCHAR2(100);
BEGIN
   SELECT instructor_id
     INTO v_instructor_id
     FROM instructor
    WHERE first_name = i_first_name
      AND last_name = i_last_name;

   SELECT COUNT(*)
     INTO v_section_count
     FROM section
```

```
    WHERE instructor_id = v_instructor_id;

   IF v_section_count >= 3 THEN
      v_status :=
         'The instructor '||i_first_name||'  '||
          i_last_name||' is teaching '||v_section_count||
         ' and needs a vaction.';
   ELSE
      v_status :=
         'The instructor '||i_first_name||'  '||
         i_last_name||' is teaching '||v_section_count||
         ' courses.';
   END IF;
   RETURN v_status;
EXCEPTION
   WHEN NO_DATA_FOUND THEN
      -- note that either of the SELECT statements can raise
      -- this exception
      v_status :=
         'The instructor '||i_first_name||'  '||
          i_last_name||' is not shown to be teaching'||
         ' any courses.';
      RETURN v_status;
   WHEN OTHERS THEN
      v_status :=
         'There has been in an error in the function.';
      RETURN v_status;
END;
```

Test the function as follows:

```
SELECT instructor_status(first_name, last_name)
  FROM instructor;
```