

# Trapping User-Defined Exceptions

## Terminology

Directions: Identify the vocabulary word for each definition below:

1. \_\_\_\_\_ A procedure used to return user-defined error messages from stored subprograms.
2. \_\_\_\_\_ Use this statement to raise a named exception.
3. \_\_\_\_\_ These errors are not automatically raised by the Oracle Server, but are defined by the programmer and are specific to the programmer's code.

## Try It / Solve It

All the questions in this exercise uses a copy of the employees table. Create this copy by running the following SQL statement:

```
CREATE TABLE excep_emps AS SELECT * FROM employees;
```

1. Create a PL/SQL block that updates the salary of every employee to a new value of 10000 in a chosen department. Include a user-defined exception handler that handles the condition where no rows are updated and displays a custom message. Also include an exception handler that will trap any other possible error condition and display the corresponding SQLCODE and SQLERRM. Test your code three times, using department\_ids 20, 30 and 40.
2. Modify your code from question 2 to handle the condition where no rows are updated using RAISE\_APPLICATION\_ERROR procedure in the exception section. Use an error number of -20202. Test your code again using department\_id 40 and check that the -20202 error is displayed.
3. Modify your code from question 3 to use RAISE\_APPLICATION\_ERROR in the executable section instead of the exception section. Test your code again using department\_id 40.

4. Before starting this question, disable Autocommit in Application Express.
- A. Enter and run the following PL/SQL block using department\_id = 40, and explain the output.

```
DECLARE
  v_dept_id  excep_emps.department_id%TYPE;
  v_count    NUMBER;
BEGIN
  v_dept_id := 40;
  SELECT COUNT(*) INTO v_count
  FROM excep_emps
  WHERE department_id = v_dept_id;
  DBMS_OUTPUT.PUT_LINE('There are ' || v_count || ' employees');
  DELETE FROM excep_emps
  WHERE department_id = v_dept_id;
  DBMS_OUTPUT.PUT_LINE(SQL%ROWCOUNT ||
    ' employees were deleted');
  ROLLBACK;
END;
```

- B. Modify your block to include two user\_defined exception handlers, one to test whether SELECT returns a value of 0, the other to test if no rows were DELETED. Declare the exceptions and RAISE them explicitly before trapping them in the exception section. Do NOT use RAISE\_APPLICATION\_ERROR. Test your modified block using department\_id 40.
- C. Modify your block again to use RAISE\_APPLICATION\_ERROR in the executable section. Use error numbers -20203 and -20204. Test your modified block using department\_id 40.