

Database Programming with PL/SQL

9-4: Managing Procedures and Functions Practice Activities

# Vocabulary

Identify the vocabulary word for each definition below:

|  |  |
| --- | --- |
| ALL\_SOURCE | The dictionary table that contains source code for all of the subprograms that you own. |
| USER\_OBJECTS | The dictionary table that contains the names and types of procedures and functions that you own. |
| USER\_SOURCE | The dictionary table that contains source code for subprograms that are owned by others who have granted you the EXECUTE privilege. |

# Try It / Solve It

1. Complete the steps below to see how exceptions are propagated.

* 1. Execute the following two SQL statements to create a duplicate of the DEPARTMENTS table, with department\_id as the primary key.

CREATE TABLE my\_depts AS SELECT \* FROM departments;

ALTER TABLE my\_depts

ADD CONSTRAINT my\_dept\_id\_pk PRIMARY KEY (department\_id);

* 1. Examine the following code and create the procedure. Save your work (you will need to modify the procedure code later).

CREATE OR REPLACE PROCEDURE add\_my\_dept

(p\_dept\_id IN VARCHAR2, p\_dept\_name IN VARCHAR2) IS

BEGIN

INSERT INTO my\_depts (department\_id, department\_name)

VALUES (p\_dept\_id, p\_dept\_name);

END add\_my\_dept;

Copyright © 2020, Oracle and/or its affiliates. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

* 1. What do you think would happen if you execute this procedure to insert department\_id 10 (which already exists)? Write and execute an anonymous block to test your theory.

BEGIN

add\_my\_dept(10, 'NewDept');

END;

**ORA-00001: unique constraint (RO\_A849\_SQL\_S15.MY\_DEPT\_ID\_PK) violated**

* 1. Modify your procedure to handle the exception in a generic WHEN OTHERS exception handler.

CREATE OR REPLACE PROCEDURE add\_my\_dept

(p\_dept\_id IN VARCHAR2, p\_dept\_name IN VARCHAR2) IS

BEGIN

INSERT INTO my\_depts (department\_id, department\_name)

VALUES (p\_dept\_id, p\_dept\_name);

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Exceptipn');

END add\_my\_dept;

* 1. Now what do you think would happen if you execute this procedure for department\_id 10 (which already exists)? Test it again as in step C.

In acest caz nu va mai aparea eroarea de mai sus, ci va aparea mesajul din exceptie.

* 1. Modify the procedure code to leave out the exception section again. Run the code.

CREATE OR REPLACE PROCEDURE add\_my\_dept

(p\_dept\_id IN VARCHAR2, p\_dept\_name IN VARCHAR2) IS BEGIN

INSERT INTO my\_depts (department\_id,department\_name) VALUES (p\_dept\_id, p\_dept\_name);

END add\_my\_dept;

* 1. Execute the following code to create a new procedure called outer\_proc which calls add\_my\_dept, passing department\_id 10 to it:

CREATE OR REPLACE PROCEDURE outer\_proc IS

v\_dept NUMBER(2) := 10; v\_dname VARCHAR2(30) := 'Admin'; BEGIN

add\_my\_dept(v\_dept, v\_dname);

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Exception was propagated to outer\_proc'); END;

* 1. Execute outer\_proc from an anonymous block. What happens and why?

BEGIN

outer\_proc;

END;

Fiindca incercam sa inseram valoarea 10, care este cheie primara, se violeaza proprietatea cheii primare si astfel are loc o exceptie. Este ridicata acea exeptie si ca urmare se va afisa mesajul din EXCEPTION.

1. Write and execute a SELECT statement to list the names of all the procedures you have created so far.

SELECT object\_name, object\_type, status

FROM user\_objects

WHERE object\_type = 'PROCEDURE';

1. Delete the last procedure you created: outer\_proc.

DROP PROCEDURE outer\_proc;

1. Write and execute a SELECT statement to list the source code of your add\_my\_dept procedure. Make sure your SELECT statement list the lines of code in the correct order.

SELECT text

FROM USER\_SOURCE

WHERE name = 'ADD\_MY\_DEPT'

ORDER BY line;