Instructor: Nebojsa Conkic  
Date: 12/30/2017

Assignment2

**DBS501**

Group Member:

Shan Zhang, #113004154

XiaoChen Wang, #015297153

Anthony Nguyen

Contents

Answer:

[Q1 2](#_Toc502405464)

[Q2 3](#_Toc502405465)

[Q2-3) 3](#_Toc502405466)

[Q2-4) 5](#_Toc502405467)

[Q5 7](#_Toc502405468)

# Q1

CREATE OR REPLACE PROCEDURE modify\_sal(

p\_dptid departments.department\_id%TYPE ) IS

deptid departments.department\_id%TYPE;

avg\_sal employees.salary%TYPE;

emp\_dpt EXCEPTION;

cursor c1 is

select department\_id, salary, first\_name, last\_name, employee\_id from employees

where department\_id = p\_dptid;

count# NUMBER(5) := 0;

diff\_sal NUMBER(5);

BEGIN

SELECT department\_id INTO deptid FROM departments WHERE department\_id = p\_dptid;

SELECT AVG(salary) INTO avg\_sal FROM employees WHERE department\_id = p\_dptid;

IF avg\_sal IS NULL THEN

RAISE emp\_dpt;

ELSE

FOR i IN c1 LOOP

IF(i.salary) < avg\_sal THEN

UPDATE employees SET salary = avg\_sal WHERE employee\_id = i.employee\_id;

count# := count# + 1;

select (avg\_sal - i.salary) into diff\_sal from dual;

dbms\_output.put\_line('Employee '|| i.first\_name ||' '|| i.last\_name ||' just got an increase of $'|| diff\_sal);

END IF;

END LOOP;

IF count# = 0 THEN

dbms\_output.put\_line('No salary was modified in Department: '||p\_dptid);

END IF;

dbms\_output.put\_line('Total # of employees who received salary increase is: ' || count#);

END IF;

EXCEPTION

WHEN no\_data\_found THEN

dbms\_output.put\_line('This Department Id is invalid: '|| p\_dptid);

WHEN emp\_dpt THEN

dbms\_output.put\_line('This Department is EMPTY: '|| p\_dptid);

END modify\_sal;

/

set serveroutput on;

set verify off;

--execute modify\_sal(99);

--execute modify\_sal(190);

--execute modify\_sal(10);

--execute modify\_sal(110);

--rollback;

execute modify\_sal(60);

rollback;

# Q2

create or replace function Total\_Cost (

f\_stuid IN student.student\_id%TYPE)

RETURN NUMBER IS

v\_stuid student.student\_id%TYPE;

totalcost NUMBER;

BEGIN

SELECT student\_id INTO v\_stuid FROM student WHERE student\_id = f\_stuid;

SELECT NVL(SUM(C.COST), 0) INTO totalcost

FROM enrollment E, section S, course C

WHERE E.section\_id = S.section\_id AND S.course\_no = C.course\_no

AND E.student\_id = f\_stuid;

IF totalcost is NULL THEN

RETURN 0;

END IF;

RETURN totalcost;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN -1;

END Total\_Cost;

/

--test 194

VARIABLE cost NUMBER

EXECUTE :cost := Total\_Cost(194)

PRINT cost

--test 294

VARIABLE cost NUMBER

EXECUTE :cost := Total\_Cost(294)

PRINT cost

--test 494

VARIABLE cost NUMBER

EXECUTE :cost := Total\_Cost(494)

PRINT cost

# Q2-3)

CREATE OR REPLACE PACKAGE My\_pack IS

PROCEDURE modify\_sal( p\_dptid departments.department\_id%TYPE ) ;

function Total\_Cost ( f\_stuid IN student.student\_id%TYPE) RETURN NUMBER;

END My\_pack;

/

CREATE OR REPLACE PACKAGE BODY My\_pack IS

--procedure

PROCEDURE modify\_sal(

p\_dptid departments.department\_id%TYPE ) IS

deptid departments.department\_id%TYPE;

avg\_sal employees.salary%TYPE;

emp\_dpt EXCEPTION;

cursor c1 is

select department\_id, salary, first\_name, last\_name, employee\_id from employees

where department\_id = p\_dptid;

count# NUMBER(5) := 0;

diff\_sal NUMBER(5);

BEGIN

SELECT department\_id INTO deptid FROM departments WHERE department\_id = p\_dptid;

SELECT AVG(salary) INTO avg\_sal FROM employees WHERE department\_id = p\_dptid;

IF avg\_sal IS NULL THEN

RAISE emp\_dpt;

ELSE

FOR i IN c1 LOOP

IF(i.salary) < avg\_sal THEN

UPDATE employees SET salary = avg\_sal WHERE employee\_id = i.employee\_id;

count# := count# + 1;

select (avg\_sal - i.salary) into diff\_sal from dual;

dbms\_output.put\_line('Employee '|| i.first\_name ||' '|| i.last\_name ||' just got an increase of $'|| diff\_sal);

END IF;

END LOOP;

IF count# = 0 THEN

dbms\_output.put\_line('No salary was modified in Department: '||p\_dptid);

END IF;

dbms\_output.put\_line('Total # of employees who received salary increase is: ' || count#);

END IF;

EXCEPTION

WHEN no\_data\_found THEN

dbms\_output.put\_line('This Department Id is invalid: '|| p\_dptid);

WHEN emp\_dpt THEN

dbms\_output.put\_line('This Department is EMPTY: '|| p\_dptid);

END modify\_sal;

--function

function Total\_Cost (

f\_stuid IN student.student\_id%TYPE)

RETURN NUMBER IS

v\_stuid student.student\_id%TYPE;

totalcost NUMBER;

BEGIN

SELECT student\_id INTO v\_stuid FROM student WHERE student\_id = f\_stuid;

SELECT NVL(SUM(C.COST), 0) INTO totalcost

FROM enrollment E, section S, course C

WHERE E.section\_id = S.section\_id AND S.course\_no = C.course\_no

AND E.student\_id = f\_stuid;

IF totalcost is NULL THEN

RETURN 0;

END IF;

RETURN totalcost;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN -1;

END Total\_Cost;

END My\_pack;

/

--test 194

VARIABLE cost NUMBER

EXECUTE :cost := My\_pack.Total\_Cost(194)

PRINT cost

--test 294

VARIABLE cost NUMBER

EXECUTE :cost := My\_pack.Total\_Cost(294)

PRINT cost

--test 494

VARIABLE cost NUMBER

EXECUTE :cost := My\_pack.Total\_Cost(494)

PRINT cost

# Q2-4)

CREATE OR REPLACE PACKAGE My\_pack IS

PROCEDURE modify\_sal( p\_dptid IN departments.department\_id%TYPE ) ;

FUNCTION Total\_Cost( f\_fname IN student.first\_name%TYPE, f\_lname IN student.last\_name%TYPE) RETURN NUMBER;

FUNCTION total\_cost ( f\_zip IN student.zip%TYPE) RETURN NUMBER;

END My\_pack;

/

CREATE OR REPLACE PACKAGE BODY My\_pack IS

--procedure

PROCEDURE modify\_sal(

p\_dptid departments.department\_id%TYPE ) IS

deptid departments.department\_id%TYPE;

avg\_sal employees.salary%TYPE;

emp\_dpt EXCEPTION;

cursor c1 is

select department\_id, salary, first\_name, last\_name, employee\_id from employees

where department\_id = p\_dptid;

count# NUMBER(5) := 0;

diff\_sal NUMBER(5);

BEGIN

SELECT department\_id INTO deptid FROM departments WHERE department\_id = p\_dptid;

SELECT AVG(salary) INTO avg\_sal FROM employees WHERE department\_id = p\_dptid;

IF avg\_sal IS NULL THEN

RAISE emp\_dpt;

ELSE

FOR i IN c1 LOOP

IF(i.salary) < avg\_sal THEN

UPDATE employees SET salary = avg\_sal WHERE employee\_id = i.employee\_id;

count# := count# + 1;

select (avg\_sal - i.salary) into diff\_sal from dual;

dbms\_output.put\_line('Employee '|| i.first\_name ||' '|| i.last\_name ||' just got an increase of $'|| diff\_sal);

END IF;

END LOOP;

IF count# = 0 THEN

dbms\_output.put\_line('No salary was modified in Department: '||p\_dptid);

END IF;

dbms\_output.put\_line('Total # of employees who received salary increase is: ' || count#);

END IF;

EXCEPTION

WHEN no\_data\_found THEN

dbms\_output.put\_line('This Department Id is invalid: '|| p\_dptid);

WHEN emp\_dpt THEN

dbms\_output.put\_line('This Department is EMPTY: '|| p\_dptid);

END modify\_sal;

--function1

function Total\_Cost (

f\_fname IN student.first\_name%TYPE,

f\_lname IN student.last\_name%TYPE)

RETURN NUMBER IS

v\_stuid student.student\_id%TYPE;

totalcost NUMBER;

BEGIN

SELECT student\_id INTO v\_stuid FROM student

WHERE upper(first\_name) = f\_fname and upper(last\_name) = f\_lname;

SELECT SUM(C.COST) INTO totalcost

FROM enrollment E, section S, course C

WHERE E.section\_id = S.section\_id AND S.course\_no = C.course\_no

AND E.student\_id = v\_stuid;

IF totalcost is NULL THEN

RETURN 0;

END IF;

RETURN totalcost;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN -1;

END Total\_Cost;

--function2

FUNCTION total\_cost (

f\_zip IN student.zip%TYPE)

RETURN NUMBER IS

v\_zipcount NUMBER;

CURSOR c1 IS

SELECT student\_id FROM student WHERE zip = f\_zip;

v\_cost NUMBER;

totalcost NUMBER := 0;

BEGIN

SELECT count(1) INTO v\_zipcount FROM student WHERE zip = f\_zip;

IF v\_zipcount = 0 THEN

raise NO\_DATA\_FOUND;

END IF;

FOR I IN c1 LOOP

--test

DBMS\_OUTPUT.PUT\_LINE(i.student\_id);

SELECT NVL(SUM(C.COST), 0) INTO v\_cost

FROM enrollment E, section S, course C

WHERE E.section\_id = S.section\_id AND S.course\_no = C.course\_no

AND E.student\_id = I.student\_id;

totalcost := totalcost + v\_cost;

END LOOP;

IF totalcost IS NULL THEN

RETURN 0;

END IF;

RETURN totalcost;

EXCEPTION

WHEN no\_data\_found THEN

RETURN -1;

END total\_cost;

END My\_pack;

/

# Q5

CREATE OR REPLACE PROCEDURE mod\_grade (

p\_courseno IN course.course\_no%TYPE,

p\_grade IN DECIMAL

)IS

v\_cno course.course\_no%TYPE;

grade\_outof\_range EXCEPTION;

count\_stu NUMBER(3);

NObody\_enroll EXCEPTION;

CURSOR c1 IS

select e.student\_id, e.section\_id

from enrollment e, section s, course c

where e.section\_id = s.section\_id and s.course\_no = c.course\_no

and c.course\_no = p\_courseno;

TotalNoGrade# NUMBER(3) := 0;

BEGIN

SELECT course\_no INTO v\_cno FROM course WHERE course\_no = p\_courseno;

IF p\_grade > 100 OR p\_grade < 0 THEN

RAISE grade\_outof\_range;

END IF;

SELECT COUNT(1) INTO count\_stu

FROM enrollment E, section S, course C

WHERE E.section\_id = S.section\_id AND S.course\_no = C.course\_no

AND C.course\_no = p\_courseno;

IF count\_stu = 0 THEN

RAISE NObody\_enroll;

END IF;

FOR i IN c1 LOOP

update enrollment set final\_grade = p\_grade

where student\_id = i.student\_id and section\_id = i.section\_id;

TotalNoGrade# := TotalNoGrade# + 1;

dbms\_output.put\_line('Student\_id: '||i.student\_id||' FinalGrade: '|| p\_grade);

END LOOP;

dbms\_output.put\_line('Total # of grades changed to '||p\_grade||' for course number '||p\_courseno||' is '||TotalNoGrade#);

EXCEPTION

WHEN no\_data\_found THEN

dbms\_output.put\_line('This Course Number is invalid: ' || p\_courseno);

WHEN grade\_outof\_range THEN

dbms\_output.put\_line('This Grade invalid: '|| p\_grade ||' It must be between 0 and 100. Try again. ');

WHEN NObody\_enroll THEN

dbms\_output.put\_line('This Course has NOBODY enrolled so far: ' || p\_courseno );

END mod\_grade;

/

SET SERVEROUTPUT ON

SET VERIFY OFF

execute mod\_grade(144, 75);

execute mod\_grade(99, 75);

execute mod\_grade(130, 101);

execute mod\_grade(130, 75);