DBS501-SSA

Assignment 2

Professor: Nebojsa Conkic

Group: 8

Group Member: Thanh Nguyen, Peng Qu, Salimeh Safi, Ori Omessi

Date: November 29th 2018

**Question 1:**

CREATE OR REPLACE PROCEDURE find\_stud

(p\_stud\_id IN student.student\_id%TYPE,

p\_last\_name OUT student.last\_name%TYPE,

p\_phone\_number OUT student.phone%TYPE,

p\_zip\_code OUT student.zip%TYPE

)

IS

BEGIN

SELECT last\_name, phone, zip INTO p\_last\_name, p\_phone\_number, p\_zip\_code

FROM student

WHERE student\_id = p\_stud\_id;

DBMS\_OUTPUT.PUT\_LINE('Student with the Id of: ' || p\_stud\_id || ' is '

|| p\_last\_name || ' with the phone# ' || p\_phone\_number ||

' and who belongs to zip code ' || p\_zip\_code);

EXCEPTION

WHEN no\_data\_found THEN

DBMS\_OUTPUT.PUT\_LINE('There is NO Student with the Id of: ' || p\_stud\_id);

END;

**OUTPUT:**

Enter SQL: var zip NUMBER

Enter SQL: var phone VARCHAR2(15)

Enter SQL: var lname VARCHAR2(30)

Enter SQL: exec find\_stud(110, :lname, :phone, :zip)

Student with the Id of: 110 is Martin with the phone# 718-555-5555 and who belongs to zip code 11385

PL/SQL procedure successfully completed.

ZIP

----------

11385

PHONE

--------------------------------

718-555-5555

LNAME

--------------------------------

Martin

Enter SQL: exec find\_stud(99, :lname, :phone, :zip)

There is NO Student with the Id of: 99

PL/SQL procedure successfully completed.

ZIP

----------

PHONE

--------------------------------

LNAME

--------------------------------

**Question 2:**

CREATE OR REPLACE PROCEDURE drop\_stud (

p\_stud\_id IN student.student\_id%TYPE,

p\_flag IN VARCHAR2 := 'R'

)

IS

p\_stud\_id\_valid student.student\_id%TYPE := NULL;

p\_stud\_enroll\_valid student.student\_id%TYPE := 0;

p\_num\_row\_deleted NUMBER(3) := 0;

no\_enrollment EXCEPTION;

BEGIN

SELECT student\_id INTO p\_stud\_id\_valid

FROM student

WHERE student\_id = p\_stud\_id;

SELECT COUNT(student\_id) INTO p\_stud\_enroll\_valid

FROM enrollment

WHERE student\_id = p\_stud\_id;

IF (p\_stud\_enroll\_valid = 0) THEN

RAISE no\_enrollment;

END IF;

IF (UPPER(p\_flag) = 'R') THEN

DBMS\_OUTPUT.PUT\_LINE('Student with the Id of : ' || p\_stud\_id || ' is enrolled in one or more courses and his/her removal is denied.');

ELSIF (UPPER(p\_flag) = 'C') THEN

DELETE FROM grade

WHERE student\_id = p\_stud\_id;

p\_num\_row\_deleted := p\_num\_row\_deleted + SQL%ROWCOUNT;

DELETE FROM enrollment

WHERE student\_id = p\_stud\_id;

p\_num\_row\_deleted := p\_num\_row\_deleted + SQL%ROWCOUNT;

DELETE FROM student

WHERE student\_id = p\_stud\_id;

p\_num\_row\_deleted := p\_num\_row\_deleted + SQL%ROWCOUNT;

DBMS\_OUTPUT.PUT\_LINE('Student with the Id of : ' || p\_stud\_id || ' is removed. Total # of rows deleted is: ' || p\_num\_row\_deleted);

END IF;

EXCEPTION

WHEN no\_data\_found THEN

DBMS\_OUTPUT.PUT\_LINE('Student with the Id of: ' || p\_stud\_id || ' does NOT exist. Try again.');

WHEN no\_enrollment THEN

DELETE FROM student

WHERE student\_id = p\_stud\_id;

DBMS\_OUTPUT.PUT\_LINE('Student with the Id of : ' || p\_stud\_id || ' is removed. He/she was NOT enrolled in any courses.');

END;

**OUTPUT:**  
case 1  
Enter SQL: exec drop\_stud(210)  
  
Student with the Id of : 210 is enrolled in one or more courses and his/her removal is denied.                                                                                                            
PL/SQL procedure successfully completed.  
Enter SQL: select section\_id, final\_grade from enrollment where student\_id = 210;  
  
  
Enter SQL: select first\_name, last\_name from student where student\_id = 210;  
FIRST\_NAME                LAST\_NAME                                                                                                                              ------------------------- ------------------------  
David                     Thares                                                                                                                                                      
case 2  
Enter SQL: exec drop\_stud(410)  
Student with the Id of: 410 does NOT exist. Try again.                                                                                                                                                    
  
PL/SQL procedure successfully completed.  
  
Enter SQL: exec drop\_stud(410,'c')  
  
Student with the Id of: 410 does NOT exist. Try again.                                                                                                                                                    
  
PL/SQL procedure successfully completed.  
  
case 3  
Enter SQL: exec drop\_stud(310)  
  
Student with the Id of : 310 is removed. He/she was NOT enrolled in any courses.                                                                                                                          
  
PL/SQL procedure successfully completed.  
  
Enter SQL: select first\_name, last\_name from student where student\_id = 310;  
no rows selected  
Enter SQL: rollback;  
Rollback complete.  
Enter SQL: select first\_name, last\_name from student where student\_id = 310;  
FIRST\_NAME                LAST\_NAME                                                                                                                               ------------------------- -------------------------   
Joseph                    Jimenes                                                                                                                                               
  
Enter SQL: exec drop\_stud(310, 'c')  
  
Student with the Id of : 310 is removed. He/she was NOT enrolled in any courses.                                                                                                                          
PL/SQL procedure successfully completed.  
Enter SQL: rollback;  
Rollback complete.

Case 4  
Enter SQL: exec drop\_stud(110)  
  
Student with the Id of : 110 is enrolled in one or more courses and his/her removal is denied.                                                                                                            
PL/SQL procedure successfully completed.  
Enter SQL: exec drop\_stud(110,'c')  
Student with the Id of : 110 is removed. Total # of rows deleted is: 24                                                                                                                                   
  
PL/SQL procedure successfully completed.  
  
Enter SQL: select section\_id, final\_grade from grade where student\_id = 110;  
  
  
no rows selected  
Enter SQL: select first\_name, last\_name from student where student\_id = 110;  
no rows selected  
Enter SQL: rollback;  
Rollback complete.  
  
Enter SQL: select section\_id, final\_grade from grade where student\_id = 110;  
  
  
Enter SQL: select first\_name, last\_name from student where student\_id = 110;  
FIRST\_NAME                LAST\_NAME                    
------------------------ -------------------------                                                                                                            
Maria                     Martin

**Question 3:**

CREATE OR REPLACE PACKAGE manage\_stud IS

PROCEDURE find\_stud(

p\_stud\_id IN student.student\_id%TYPE,

p\_last\_name OUT student.last\_name%TYPE,

p\_phone\_number OUT student.phone%TYPE,

p\_zip\_code OUT student.zip%TYPE

);

PROCEDURE drop\_stud(

p\_stud\_id IN student.student\_id%TYPE,

p\_flag IN VARCHAR2 := 'R'

);

END manage\_stud;

CREATE OR REPLACE PACKAGE BODY manage\_stud IS

PROCEDURE find\_stud(

p\_stud\_id IN student.student\_id%TYPE,

p\_last\_name OUT student.last\_name%TYPE,

p\_phone\_number OUT student.phone%TYPE,

p\_zip\_code OUT student.zip%TYPE

) IS

BEGIN

SELECT last\_name, phone, zip INTO p\_last\_name, p\_phone\_number, p\_zip\_code

FROM student

WHERE student\_id = p\_stud\_id;

DBMS\_OUTPUT.PUT\_LINE('Student with the Id of: ' || p\_stud\_id || ' is '

|| p\_last\_name || ' with the phone# ' || p\_phone\_number ||

' and who belongs to zip code ' || p\_zip\_code);

EXCEPTION

WHEN no\_data\_found THEN

DBMS\_OUTPUT.PUT\_LINE('There is NO Student with the Id of: ' || p\_stud\_id);

END find\_stud;

PROCEDURE drop\_stud(

p\_stud\_id IN student.student\_id%TYPE,

p\_flag IN VARCHAR2 := 'R'

) IS

p\_stud\_id\_valid student.student\_id%TYPE := NULL;

p\_stud\_enroll\_valid student.student\_id%TYPE := 0;

p\_num\_row\_deleted NUMBER(3) := 0;

no\_enrollment EXCEPTION;

BEGIN

SELECT student\_id INTO p\_stud\_id\_valid

FROM student

WHERE student\_id = p\_stud\_id;

SELECT COUNT(student\_id) INTO p\_stud\_enroll\_valid

FROM enrollment

WHERE student\_id = p\_stud\_id;

IF (p\_stud\_enroll\_valid = 0) THEN

RAISE no\_enrollment;

END IF;

IF (UPPER(p\_flag) = 'R') THEN

DBMS\_OUTPUT.PUT\_LINE('Student with the Id of : ' || p\_stud\_id || ' is enrolled in one or more courses and his/her removal is denied.');

ELSIF (UPPER(p\_flag) = 'C') THEN

DELETE FROM grade

WHERE student\_id = p\_stud\_id;

p\_num\_row\_deleted := p\_num\_row\_deleted + SQL%ROWCOUNT;

DELETE FROM enrollment

WHERE student\_id = p\_stud\_id;

p\_num\_row\_deleted := p\_num\_row\_deleted + SQL%ROWCOUNT;

DELETE FROM student

WHERE student\_id = p\_stud\_id;

p\_num\_row\_deleted := p\_num\_row\_deleted + SQL%ROWCOUNT;

DBMS\_OUTPUT.PUT\_LINE('Student with the Id of : ' || p\_stud\_id || ' is removed. Total # of rows deleted is: ' || p\_num\_row\_deleted);

END IF;

EXCEPTION

WHEN no\_data\_found THEN

DBMS\_OUTPUT.PUT\_LINE('Student with the Id of: ' || p\_stud\_id || ' does NOT exist. Try again.');

WHEN no\_enrollment THEN

DELETE FROM student

WHERE student\_id = p\_stud\_id;

DBMS\_OUTPUT.PUT\_LINE('Student with the Id of : ' || p\_stud\_id || ' is removed. He/she was NOT enrolled in any courses.');

END drop\_stud;

END manage\_stud;

**OUTPUT:**

Enter SQL: var zip NUMBER

Enter SQL: var phone VARCHAR2(15)

Enter SQL: var lname VARCHAR2(30)

Enter SQL: exec manage\_stud.find\_stud(110, :lname,:phone,:zip)

Student with the Id of: 110 is Martin with the phone# 718-555-5555 and who belongs to zip code 11385

PL/SQL procedure successfully completed.

ZIP

----------

11385

PHONE

--------------------------------

718-555-5555

LNAME

--------------------------------

Martin

Enter SQL: exec manage\_stud.drop\_stud(310)

Student with the Id of : 310 is removed. He/she was NOT enrolled in any courses.

PL/SQL procedure successfully completed.

Enter SQL: rollback;

Rollback complete.

Enter SQL: exec manage\_stud.drop\_stud(110,'C')

Student with the Id of : 110 is removed. Total # of rows deleted is: 24

PL/SQL procedure successfully completed.

Enter SQL: rollback;

Rollback complete.

**Question 4:**

CREATE OR REPLACE PACKAGE manage\_stud IS

PROCEDURE find\_stud(

p\_stud\_id IN student.student\_id%TYPE,

p\_last\_name OUT student.last\_name%TYPE,

p\_phone\_number OUT student.phone%TYPE,

p\_zip\_code OUT student.zip%TYPE

);

-- Overloaded Procedure

**PROCEDURE find\_stud(**

**p\_stud\_id IN student.student\_id%TYPE,**

**p\_first\_name OUT student.first\_name%TYPE,**

**p\_last\_name OUT student.last\_name%TYPE**

**);**

PROCEDURE drop\_stud(

p\_stud\_id IN student.student\_id%TYPE,

p\_flag IN VARCHAR2 := 'R'

);

-- New Function

**FUNCTION valid\_stud (**

**f\_stud\_id student.student\_id%TYPE**

**) RETURN BOOLEAN;**

END manage\_stud;

CREATE OR REPLACE PACKAGE BODY manage\_stud IS

PROCEDURE find\_stud(

p\_stud\_id IN student.student\_id%TYPE,

p\_last\_name OUT student.last\_name%TYPE,

p\_phone\_number OUT student.phone%TYPE,

p\_zip\_code OUT student.zip%TYPE

) IS

BEGIN

SELECT last\_name, phone, zip INTO p\_last\_name, p\_phone\_number, p\_zip\_code

FROM student

WHERE student\_id = p\_stud\_id;

DBMS\_OUTPUT.PUT\_LINE('Student with the Id of: ' || p\_stud\_id || ' is '

|| p\_last\_name || ' with the phone# ' || p\_phone\_number ||

' and who belongs to zip code ' || p\_zip\_code);

EXCEPTION

WHEN no\_data\_found THEN

DBMS\_OUTPUT.PUT\_LINE('There is NO Student with the Id of: ' || p\_stud\_id);

END find\_stud;

**PROCEDURE find\_stud(**

**p\_stud\_id IN student.student\_id%TYPE,**

**p\_first\_name OUT student.first\_name%TYPE,**

**p\_last\_name OUT student.last\_name%TYPE**

**) IS**

**BEGIN**

**IF (valid\_stud(p\_stud\_id) = TRUE) THEN**

**SELECT last\_name, first\_name INTO p\_last\_name, p\_first\_name**

**FROM student**

**WHERE student\_id = p\_stud\_id;**

**DBMS\_OUTPUT.PUT\_LINE('Student with the Id of: ' || p\_stud\_id || ' is '**

**|| p\_first\_name || ' ' || p\_last\_name);**

**ELSE**

**DBMS\_OUTPUT.PUT\_LINE('There is NO Student with the Id of: ' || p\_stud\_id);**

**END IF;**

**END find\_stud;**

PROCEDURE drop\_stud(

p\_stud\_id IN student.student\_id%TYPE,

p\_flag IN VARCHAR2 := 'R'

) IS

p\_stud\_id\_valid student.student\_id%TYPE := NULL;

p\_stud\_enroll\_valid student.student\_id%TYPE := 0;

p\_num\_row\_deleted NUMBER(3) := 0;

no\_enrollment EXCEPTION;

BEGIN

SELECT student\_id INTO p\_stud\_id\_valid

FROM student

WHERE student\_id = p\_stud\_id;

SELECT COUNT(student\_id) INTO p\_stud\_enroll\_valid

FROM enrollment

WHERE student\_id = p\_stud\_id;

IF (p\_stud\_enroll\_valid = 0) THEN

RAISE no\_enrollment;

END IF;

IF (UPPER(p\_flag) = 'R') THEN

DBMS\_OUTPUT.PUT\_LINE('Student with the Id of : ' || p\_stud\_id || ' is enrolled in one or more courses and his/her removal is denied.');

ELSIF (UPPER(p\_flag) = 'C') THEN

DELETE FROM grade

WHERE student\_id = p\_stud\_id;

p\_num\_row\_deleted := p\_num\_row\_deleted + SQL%ROWCOUNT;

DELETE FROM enrollment

WHERE student\_id = p\_stud\_id;

p\_num\_row\_deleted := p\_num\_row\_deleted + SQL%ROWCOUNT;

DELETE FROM student

WHERE student\_id = p\_stud\_id;

p\_num\_row\_deleted := p\_num\_row\_deleted + SQL%ROWCOUNT;

DBMS\_OUTPUT.PUT\_LINE('Student with the Id of : ' || p\_stud\_id || ' is removed. Total # of rows deleted is: ' || p\_num\_row\_deleted);

END IF;

EXCEPTION

WHEN no\_data\_found THEN

DBMS\_OUTPUT.PUT\_LINE('Student with the Id of: ' || p\_stud\_id || ' does NOT exist. Try again.');

WHEN no\_enrollment THEN

DELETE FROM student

WHERE student\_id = p\_stud\_id;

DBMS\_OUTPUT.PUT\_LINE('Student with the Id of : ' || p\_stud\_id || ' is removed. He/she was NOT enrolled in any courses.');

END drop\_stud;

**FUNCTION valid\_stud (**

**f\_stud\_id student.student\_id%TYPE**

**) RETURN BOOLEAN**

**IS**

**f\_stud\_dummy student.student\_id%TYPE;**

**BEGIN**

**SELECT student\_id INTO f\_stud\_dummy**

**FROM student**

**WHERE student\_id = f\_stud\_id;**

**RETURN TRUE;**

**EXCEPTION**

**WHEN no\_data\_found THEN**

**RETURN FALSE;**

**END valid\_stud;**

END manage\_stud;

**OUTPUT:**

Enter SQL: var fname VARCHAR2(30)

Enter SQL: var lname VARCHAR2(30)

Enter SQL: exec manage\_stud.find\_stud(110, :fname, :lname)

Student with the Id of: 110 is Maria Martin

PL/SQL procedure successfully completed.

LNAME

--------------------------------

Martin

FNAME

--------------------------------

Maria

**Question 5:**

ALTER TABLE countries

ADD FLAG CHAR(7)

/

CREATE OR REPLACE PROCEDURE question5 (

p\_region\_id regions.region\_id%TYPE

)

IS

CURSOR p\_c\_countries IS

SELECT country\_id, country\_name

FROM countries

ORDER BY country\_name;

TYPE country\_table IS TABLE OF countries.country\_name%TYPE

INDEX BY PLS\_INTEGER;

p\_country\_table country\_table;

p\_region\_exist NUMBER(1);

p\_country\_empty NUMBER(3);

p\_table\_index NUMBER(3) := 1;

p\_country\_no\_city NUMBER(1);

p\_counter NUMBER(3);

p\_num\_country\_with\_no\_city NUMBER(3) := 0;

BEGIN

SELECT COUNT(region\_id) INTO p\_region\_exist

FROM regions

WHERE region\_id = p\_region\_id;

IF (p\_region\_exist = 0) THEN

DBMS\_OUTPUT.PUT\_LINE('This region ID does NOT exist: ' || p\_region\_id);

ELSE

FOR country\_rec IN p\_c\_countries LOOP

SELECT COUNT(country\_id) INTO p\_country\_empty

FROM locations

WHERE country\_id = country\_rec.country\_id;

IF (p\_country\_empty = 0) THEN

UPDATE countries

SET flag = 'EMPTY\_' || region\_id

WHERE country\_id = country\_rec.country\_id;

SELECT country\_name INTO p\_country\_table(p\_table\_index)

FROM countries

WHERE country\_id = country\_rec.country\_id;

DBMS\_OUTPUT.PUT\_LINE('Index Table Key: ' || p\_table\_index || ' has a value of ' || p\_country\_table(p\_table\_index));

p\_table\_index := p\_table\_index + 5;

END IF;

END LOOP;

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

DBMS\_OUTPUT.PUT\_LINE('Total number of elements in the Index Table or Number of countries with NO cities listed is: ' || p\_country\_table.COUNT);

DBMS\_OUTPUT.PUT\_LINE('Second element (Country) in the Index Table is: ' || p\_country\_table(p\_country\_table.NEXT(p\_country\_table.FIRST)));

DBMS\_OUTPUT.PUT\_LINE('Before the last element (Country) in the Index Table is: ' || p\_country\_table(p\_country\_table.PRIOR(p\_country\_table.LAST)));

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

p\_table\_index := 1;

FOR p\_counter IN 1..p\_country\_table.COUNT LOOP

SELECT COUNT(country\_name) INTO p\_country\_no\_city

FROM countries

WHERE country\_name = p\_country\_table(p\_table\_index) AND region\_id = p\_region\_id;

-- Found

IF (p\_country\_no\_city = 1) THEN

p\_num\_country\_with\_no\_city := p\_num\_country\_with\_no\_city + 1;

DBMS\_OUTPUT.PUT\_LINE('In the region ' || p\_region\_id || ' there is country ' || p\_country\_table(p\_table\_index) || ' with NO city.');

END IF;

p\_table\_index := p\_table\_index + 5;

END LOOP;

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

DBMS\_OUTPUT.PUT\_LINE('Total Number of countries with NO cities listed in the Region ' || p\_region\_id || ' is: ' || p\_num\_country\_with\_no\_city);

END IF;

END;

**OUTPUT:**

Enter SQL: ALTER TABLE countries

2 ADD FLAG CHAR(7)

3 /

Table altered.

Enter SQL: ACCEPT region PROMPT 'Enter value for region: '

Enter value for region: 5

Enter SQL: exec question5(&region)

This region ID does NOT exist: 5

PL/SQL procedure successfully completed.

Enter SQL: SELECT \* FROM COUNTRIES

2 WHERE flag IS NOT NULL;

no rows selected

Enter SQL: rollback;

Rollback complete.

Enter SQL: ACCEPT region PROMPT 'Enter value for region: '

Enter value for region: 1

Enter SQL: exec question5(&region)

Index Table Key: 1 has a value of Argentina

Index Table Key: 6 has a value of Belgium

Index Table Key: 11 has a value of Denmark

Index Table Key: 16 has a value of Egypt

Index Table Key: 21 has a value of France

Index Table Key: 26 has a value of HongKong

Index Table Key: 31 has a value of Israel

Index Table Key: 36 has a value of Kuwait

Index Table Key: 41 has a value of Nigeria

Index Table Key: 46 has a value of Zambia

Index Table Key: 51 has a value of Zimbabwe

======================================================================

Total number of elements in the Index Table or Number of countries with NO cities listed is: 11

Second element (Country) in the Index Table is: Belgium

Before the last element (Country) in the Index Table is: Zambia

======================================================================

In the region 1 there is country Belgium with NO city.

In the region 1 there is country Denmark with NO city.

In the region 1 there is country France with NO city.

======================================================================

Total Number of countries with NO cities listed in the Region 1 is: 3

PL/SQL procedure successfully completed.

Enter SQL: SELECT \* FROM COUNTRIES

2 WHERE flag IS NOT NULL;

CO COUNTRY\_NAME REGION\_ID FLAG

-- ---------------------------------------- ---------- -------

AR Argentina 2 EMPTY\_2

BE Belgium 1 EMPTY\_1

DK Denmark 1 EMPTY\_1

EG Egypt 4 EMPTY\_4

FR France 1 EMPTY\_1

HK HongKong 3 EMPTY\_3

IL Israel 4 EMPTY\_4

KW Kuwait 4 EMPTY\_4

NG Nigeria 4 EMPTY\_4

ZM Zambia 4 EMPTY\_4

ZW Zimbabwe 4 EMPTY\_4

11 rows selected.

Enter SQL: rollback;

Rollback complete.

Enter SQL: ACCEPT region PROMPT 'Enter value for region: '

Enter value for region: 2

Enter SQL: exec question5(&region)

Index Table Key: 1 has a value of Argentina

Index Table Key: 6 has a value of Belgium

Index Table Key: 11 has a value of Denmark

Index Table Key: 16 has a value of Egypt

Index Table Key: 21 has a value of France

Index Table Key: 26 has a value of HongKong

Index Table Key: 31 has a value of Israel

Index Table Key: 36 has a value of Kuwait

Index Table Key: 41 has a value of Nigeria

Index Table Key: 46 has a value of Zambia

Index Table Key: 51 has a value of Zimbabwe

======================================================================

Total number of elements in the Index Table or Number of countries with NO cities listed is: 11

Second element (Country) in the Index Table is: Belgium

Before the last element (Country) in the Index Table is: Zambia

======================================================================

In the region 2 there is country Argentina with NO city.

======================================================================

Total Number of countries with NO cities listed in the Region 2 is: 1

PL/SQL procedure successfully completed.

Enter SQL: SELECT \* FROM COUNTRIES

2 WHERE flag IS NOT NULL;

CO COUNTRY\_NAME REGION\_ID FLAG

-- ---------------------------------------- ---------- -------

AR Argentina 2 EMPTY\_2

BE Belgium 1 EMPTY\_1

DK Denmark 1 EMPTY\_1

EG Egypt 4 EMPTY\_4

FR France 1 EMPTY\_1

HK HongKong 3 EMPTY\_3

IL Israel 4 EMPTY\_4

KW Kuwait 4 EMPTY\_4

NG Nigeria 4 EMPTY\_4

ZM Zambia 4 EMPTY\_4

ZW Zimbabwe 4 EMPTY\_4

11 rows selected.

Enter SQL: rollback;

Rollback complete.