

דו"ח שלב 3

מגישות: תהילה בן משה - 213385263

מרים פרץ - 322241100

פונקציות:

פונקציה מס' 1

תיאור: בהינתן מזהה מדריך, נרצה לחשב את הבונוס שיקבל לאחר סיום הקורס:

כמות ההרשמה לקורס שמלמד * מחיר הקורס * 10%

הקוד:

```
1 create or replace function ReturnInstructorSalary(instructorID in number) return number is
2     Result NUMBER := 0;
3     registrationCount INTEGER;
4     CURSOR courseCursor IS
5         SELECT C.Course_ID, C.price
6         FROM Course C
7         WHERE C.Instructor_ID = instructorID;
8     courseRecord courseCursor%ROWTYPE;
9
10 BEGIN
11     -- Open the cursor
12     OPEN courseCursor;
13
14     -- Loop through each course taught by the instructor
15     LOOP
16         -- Fetch the next course
17         FETCH courseCursor INTO courseRecord;
18         EXIT WHEN courseCursor%NOTFOUND;
19
20         -- Get the number of registrations for the current course
21         BEGIN
22             SELECT COUNT(*)
23             INTO registrationCount
24             FROM Registration R
25             WHERE R.Course_ID = courseRecord.Course_ID;
26         EXCEPTION
27             WHEN NO_DATA_FOUND THEN
28                 registrationCount := 0;
29         END;
30
31         -- Calculate the salary contribution from the current course
32         Result := Result + (registrationCount * courseRecord.price * 0.1);
33     END LOOP;
34
35     -- Close the cursor
36     CLOSE courseCursor;
37
38     -- Return the calculated salary
39     RETURN Result;
40 END ReturnInstructorSalary;
```








דוגמא:

לצורך הדוגמא ניקח את המדריך עם ת"ז = 113110911 ובקורס אצלו יש 6 משתתפים

```
--help query for function 1
SELECT DISTINCT I.Instructor_ID, I.fname, I.lname, C.Course_Name, C.PRICE
FROM Instructor I
JOIN Course C ON I.Instructor_ID = C.Instructor_ID
JOIN Registration R ON C.Course_ID = R.Course_ID
GROUP BY I.Instructor_ID, I.fname, I.lname, C.Course_Name, C.PRICE
HAVING COUNT(DISTINCT R.Participant_ID) = 6;
```

	INSTRUCTOR_ID	FNAME	LNAME	COURSE_NAME	PRICE
1	113110911	Dave	Mratz	First Aid for Animal Bite ...	373

Test script DBMS Output Statistics Profiler Trace

```
1 begin
2   -- Call the function
3   :result := returninstructorsalary(instructorid => :instruct
4 end;
```

	Variable	Type	Value
▶	result	Float	223.8
	instructorid	Float	113110911
*			

פונקציה מס' 2

תיאור:

בהינתן שם של מחסן קטגוריה (category location) נרצה לחשב על הערך המשוער של המחסן. הערך המשוער מחושב כך:

סכום האביזרים במחסן * מספר הקטגוריות הנמצאות בו / המשקל הממוצע של האביזרים המאוחסנים במחסן.

הקוד:

```
CREATE OR REPLACE FUNCTION Calculate_Location_Value (p_category_location VARCHAR2)
RETURN NUMBER
IS
  -- Declare variables and types
  TYPE tool_rec IS RECORD (
    Tool_name Tool.Tool_name%TYPE,
    weight Tool.weight%TYPE
  );

  TYPE tool_cursor IS REF CURSOR;

  v_tool tool_rec;
  v_tool_cursor tool_cursor;

  v_total_weight FLOAT := 0;
  v_tool_count NUMBER := 0;
  v_category_count NUMBER := 0;
  v_avg_weight FLOAT := 0;
  v_location_value FLOAT := 0;

  -- Declare implicit cursor
  CURSOR category_cursor IS
    SELECT Category_name
    FROM TCategory
    WHERE Category_location = p_category_location;

BEGIN
  -- Get the number of categories in the location
  OPEN category_cursor;
  LOOP
    FETCH category_cursor INTO v_tool.Tool_name;
    EXIT WHEN category_cursor%NOTFOUND;
    v_category_count := v_category_count + 1;

    -- Explicit cursor to get tools for each category
    OPEN v_tool_cursor FOR
      SELECT Tool_name, weight
      FROM Tool
      WHERE Category_name = v_tool.Tool_name;

    LOOP
      FETCH v_tool_cursor INTO v_tool;
      EXIT WHEN v_tool_cursor%NOTFOUND;
      v_total_weight := v_total_weight + v_tool.weight;
      v_tool_count := v_tool_count + 1;
    END LOOP;

    CLOSE v_tool_cursor;
  END LOOP;

  CLOSE category_cursor;

  -- Calculate the average weight
  IF v_tool_count > 0 THEN
    v_avg_weight := v_total_weight / v_tool_count;
  ELSE
    v_avg_weight := 0;
  END IF;

  -- Calculate the location value
  IF v_avg_weight > 0 THEN
    v_location_value := (v_category_count * v_tool_count) / v_avg_weight;
  ELSE
    v_location_value := 0;
  END IF;
END;
```

```

-- Return the calculated location value
RETURN v_location_value;

EXCEPTION
WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('No data found for the given category location. ');
    RETURN 0;
WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
    RETURN 0;
END Calculate_Location_Value;

```

דוגמא:

לצורך הדוגמא ניקח את המחסן 'Storage room 1'
נחשב את הערך של המחסן בפונקציה:

DBMS Output			
1	begin		
2	-- Call the function		
3	:result := calculate_location_value(p_category_location => :p_category_location);		
4	end;		

<input type="checkbox"/>	Variable	Type	Value
<input checked="" type="checkbox"/>	result	Float	6421.99488491048
<input checked="" type="checkbox"/>	p_category_location	String	Storage Room 1
<input checked="" type="checkbox"/>	*		

לצורך בדיקה, נכתוב 3 שאילתות. אחת תחזיר את כמות האביזרים במחסן, אחת את מספר הקטגוריות בו ואחת את המשקל הממוצע:

```

SELECT COUNT(*) AS number_of_categories
FROM TCategory
WHERE Category_location = 'Storage Room 1';

```

NUMBER_OF_CATEGORIES	
1	31

```

SELECT COUNT(*) AS number_of_tools
FROM Tool t
JOIN TCategory c ON t.Category_name = c.Category_name
WHERE c.Category_location = 'Storage Room 1';

```

NUMBER_OF_TOOLS	
1	90

```
SELECT AVG(t.weight) AS average_weight  
FROM Tool t  
JOIN TCategory c ON t.Category_name = c.Category_name  
WHERE c.Category_location = 'Storage Room 1';
```



	AVERAGE_WEIGHT
1	0.434444444444444

נחשב לפי הנוסחה שלעיל ונקבל מספר השווה לתוצאת הפונקציה שלנו.

פרוצדורות:

פרוצדורה 1

תיאור:

נרצה לבנות פרוצדורה שתעביר את כל המשתתפים מקורס אחד לשני עקב סגירת הקורס או העברתו למקום אחר. הפרוצדורה תעדכן את ההרשמה של המשתתפים בהתאם וכן תוודא שאין רישום כפול של המשתתף לקורס אליו הוא מועבר.

הקוד:

```
CREATE OR REPLACE PROCEDURE TransferParticipants(
    from_course_id IN Course.Course_ID%TYPE,
    to_course_id IN Course.Course_ID%TYPE
)
IS
    TYPE ParticipantRec IS RECORD (
        participant_id Participant.Participant_ID%TYPE,
        registration_id Registration.Registration_ID%TYPE
    );

    CURSOR participant_cursor IS
        SELECT p.Participant_ID, r.Registration_ID
        FROM Participant p
        JOIN Registration r ON p.Participant_ID = r.Participant_ID
        WHERE r.Course_ID = from_course_id;

    participant_rec ParticipantRec;
    err_msg VARCHAR2(32767);

BEGIN
    OPEN participant_cursor;
    LOOP
        FETCH participant_cursor INTO participant_rec;
        EXIT WHEN participant_cursor%NOTFOUND;

        -- Check if participant is already registered for the to_course
        DECLARE
            duplicate_count INT;
        BEGIN
            SELECT COUNT(*)
            INTO duplicate_count
            FROM Registration
            WHERE Participant_ID = participant_rec.participant_id
            AND Course_ID = to_course_id;

            IF duplicate_count = 0 THEN
                -- Update registration
                UPDATE Registration
                SET Course_ID = to_course_id
                WHERE Registration_ID = participant_rec.registration_id;
            ELSE
                -- Skip this participant
                DBMS_OUTPUT.PUT_LINE('Participant ' || participant_rec.participant_id || ' is already registered');
            END IF;
        END;
    END LOOP;
    CLOSE participant_cursor;

    OPEN ref_cur FOR SELECT 'Participants transferred successfully' AS message FROM dual;
    COMMIT;

EXCEPTION
    WHEN OTHERS THEN
        err_msg := 'Error occurred: ' || TO_CHAR(SQLCODE) || ' - ' || SUBSTR(SQLERRM, 1, 200);
        OPEN ref_cur FOR SELECT err_msg AS message FROM dual;
        ROLLBACK;

END;
```

בדיקה:

לצורך הבדיקה, מתוך רשימת הקורסים נמצא שני קורסים עם אותו שם- ניקח את CPR אך עם מיקום שונה ואותם נרצה לאחד:

	COURSE_ID	COURSE_NAME	COURSE_DATE	COURSE_LOCATION	PRICE	HOURS	INSTRUCTOR_ID
15	1357	Advanced Cardiac Life Sup ...	4/15/2024	Netanya	3630	22	113842545
6	1459	Advanced Cardiac Life Sup ...	11/5/2024	Hadera	3300	9	111110674
9	1470	Basic First Aid	2/14/2024	Tel Aviv	3488	16	101889670
12	1356	CPR	1/24/2024	Hadera	2493	10	103882372
18	1018	CPR	5/21/2024	Ashkelon	2478	1	110061477

כמות המשתתפים שיש בקורס הראשון '1356':

```
SELECT P.Participant_ID, P.fname, P.lname, P.address, P.email, P.birth_date
FROM Participant P
JOIN Registration R ON P.Participant_ID = R.Participant_ID
WHERE R.Course_ID = '1356';
```

	PARTICIPANT_ID	FNAME	LNAME	ADDRESS	EMAIL	BIRTH_DATE
1	631311297	Ozzy	Roj	983 Oakridge Avenue	troj4n@jigsy.com	12/27/2003
2	883547761	Gannon	Matevushev	545 Transport Trail	rmatevushevb2@bandcamp.com	3/31/1966
3	967655547	Ingaborg	Bedberry	3 Grayhawk Terrace	lbedberry9@google.co.jp	5/14/1963

בקורס השני '1018':

```
SELECT P.Participant_ID, P.fname, P.lname, P.address, P.email, P.birth_date
FROM Participant P
JOIN Registration R ON P.Participant_ID = R.Participant_ID
WHERE R.Course_ID = '1018';
```

	PARTICIPANT_ID	FNAME	LNAME	ADDRESS	EMAIL	BIRTH_DATE
1	817295135	Fernando	McIlmorie	34220 Corry Road	wmcilmorie2h@wisc.edu	3/30/1961
2	208092458	Marcille	Edgeler	538 Village Pass	vedgeler5p@elpais.com	8/8/2003
3	445415422	Nydia	Hune	69813 Transport Avenue	khuneq@cnbc.com	5/27/1960

Test script DBMS Output Statistics Profiler Trace

```

1 begin
2     -- Call the procedure
3     transferparticipants(from_course_id => :from_course_id,
4                         to_course_id => :to_course_id,
5                         ref_cur => :ref_cur);
6 end;

```

Variable	Type	Value
from_course_id	Integer	1356
to_course_id	Integer	1018
ref_cur	Cursor	<Cursor>

לאחר הפעלת הפרוצדורה,
בקורס הראשון ניתן לראות שאין משתתפים:

```

----- 1356 1018
SELECT P.Participant_ID, P.fname, P.lname, P.address, P.email, P.birth_date
FROM Participant P
JOIN Registration R ON P.Participant_ID = R.Participant_ID
WHERE R.Course_ID = '1356';

```

PARTICIPANT_ID	FNAME	LNAME	ADDRESS	EMAIL	BIRTH_DATE
----------------	-------	-------	---------	-------	------------

ואכן המשתתפים עברו לקורס השני:

```

----- 1356 1018
SELECT P.Participant_ID, P.fname, P.lname, P.address, P.email, P.birth_date
FROM Participant P
JOIN Registration R ON P.Participant_ID = R.Participant_ID
WHERE R.Course_ID = '1018';

```

PARTICIPANT_ID	FNAME	LNAME	ADDRESS	EMAIL	BIRTH_DATE
1	Ozy	Roj	983 Oakridge Avenue	troj4n@jigsy.com	27/12/2003
2	Fernando	McIlmorie	34220 Corry Road	wmcilmorie2h@wisc.edu	30/03/1961
3	Marcille	Edgeler	538 Village Pass	vedgeler5p@elpais.com	08/08/2003
4	Nydia	Hune	69813 Transport Avenue	khuneq@cnbc.com	27/05/1960
5	Gannon	Matevushev	545 Transport Trail	rmatevushev2@bandcamp.com	31/03/1966
6	Ingaborg	Bedberry	3 Grayhawk Terrace	lbedberry9@google.co.jp	14/05/1963

פרוצדורה 2

תיאור:

בהיתן מזהה מדריך שידוע על הצטיינותו, נרצה לבנות פרוצדורה שתחזיר סיכום מלא עם פרטיו. הפרטים כוללים: שמות ותאריכי הקורסים שמלמד וכן המשתתפים בכל קורס. הפרוצדורה תוסיף טיפול בחריגות במקרה שבו מוכנס מזהה שגוי. בנוסף, הפרוצדורה תעדכן בבסיס הנתונים את הסמכת המדריך - תוסיף את המילה "excellent" לצד הסמכת הרגילה.

הקוד:

```
1 CREATE OR REPLACE PROCEDURE InstructorSummary(InstructorID IN NUMBER) IS
2   -- Cursor to fetch the courses taught by the instructor
3   CURSOR courses_cur IS
4     SELECT c.Course_ID, c.Course_name, c.Course_date
5     FROM Course c
6     WHERE c.Instructor_ID = InstructorID
7     ORDER BY c.Course_date DESC;
8
9   -- Variable to hold each fetched row from courses_cur
10  course_record courses_cur%ROWTYPE;
11
12  -- Variables to hold instructor details
13  v_instructor_fname Instructor.fname%TYPE;
14  v_instructor_lname Instructor.lname%TYPE;
15  v_instructor_qualification Instructor.qualification%TYPE;
16
17  -- Variable to check if instructor exists
18  v_instructor_exists NUMBER := 0;
19 BEGIN
20   -- Increase the buffer size to 1,000,000 bytes
21   DBMS_OUTPUT.ENABLE(1000000);
22
23   -- Fetch instructor's name and qualification
24   SELECT fname, lname, qualification INTO v_instructor_fname, v_instructor_lname, v_instructor_qualification
25   FROM Instructor
26   WHERE Instructor_ID = InstructorID;
27
28   -- Check if instructor exists
29   SELECT COUNT(*)
30   INTO v_instructor_exists
31   FROM Instructor
32   WHERE Instructor_ID = InstructorID;
33
34   IF v_instructor_exists = 0 THEN
35     DBMS_OUTPUT.PUT_LINE('Instructor with ID ' || InstructorID || ' does not exist.');
```

```
36     RETURN;
37   END IF;
38
39   -- Enhance qualification with "excellent"
40   v_instructor_qualification := v_instructor_qualification || ' excellent';
41
42   -- Update instructor's qualification in the database
43   UPDATE Instructor
44   SET qualification = v_instructor_qualification
45   WHERE Instructor_ID = InstructorID;
46
47   -- Commit the update
48   COMMIT;
49
50   DBMS_OUTPUT.PUT_LINE('-----');
51   DBMS_OUTPUT.PUT_LINE(' Instructor Summary');
52   DBMS_OUTPUT.PUT_LINE('-----');
53   DBMS_OUTPUT.PUT_LINE('Instructor: ' || v_instructor_fname || ' ' || v_instructor_lname);
54   DBMS_OUTPUT.PUT_LINE('Qualification: ' || v_instructor_qualification);
55   DBMS_OUTPUT.PUT_LINE('');
```

```

-- Open the cursor for courses
OPEN courses_cur;
LOOP
    FETCH courses_cur INTO course_record;
    EXIT WHEN courses_cur%NOTFOUND;

    -- Print course details
    DBMS_OUTPUT.PUT_LINE('Course Name: ' || course_record.Course_name);
    DBMS_OUTPUT.PUT_LINE('Course Date: ' || TO_CHAR(course_record.Course_date, 'DD-MON-YYYY'));
    DBMS_OUTPUT.PUT_LINE('Participants:');

    -- Cursor to fetch the participants for the current course
    FOR participant_record IN (
        SELECT p.fname, p.lname
        FROM Participant p
        JOIN Registration r ON r.Participant_ID = p.Participant_ID
        WHERE r.Course_ID = course_record.Course_ID
        ORDER BY p.lname, p.fname
    ) LOOP
        -- Print participant details
        DBMS_OUTPUT.PUT_LINE(' - ' || participant_record.fname || ' ' || participant_record.lname);
    END LOOP;

    DBMS_OUTPUT.PUT_LINE('');
END LOOP;
CLOSE courses_cur;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('No data found for this Instructor ' || InstructorID);
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
END InstructorSummary;

```

בדיקה

Test script	DBMS Output	Statistics	Profiler	Trace						
<pre> 1 begin 2 -- Call the procedure 3 instructorsummary(instructorid => :instructorid); 4 end; </pre>										
	<table border="1"> <thead> <tr> <th>Variable</th> <th>Type</th> <th>Value</th> </tr> </thead> <tbody> <tr> <td>instructorid</td> <td>Float</td> <td>114137866</td> </tr> </tbody> </table>	Variable	Type	Value	instructorid	Float	114137866			
Variable	Type	Value								
instructorid	Float	114137866								

Test script	DBMS Output	Statistics	Profiler	Trace
<div>Clear</div> <div>Buffer size 10000</div> <div>Enabled</div>	<pre> ----- Instructor Summary ----- Instructor: Brittany Loggins Qualification: nurse excellent Course Name: First Aid for Hypothermia Course Date: 15-AUG-2024 Participants: - Glyn Croucher - Rich Fettis - Dun Tonna </pre>			

נוודא שהפרטים עבור המדריך שהכנסנו נכונים. לצורך כך נכתוב שאילתא שתחזיר את פרטי המדריך עבור המזהה שהכנסנו לעיל (כולל העדכון שנעשה בבסיס הנתונים):

```
SELECT
i.fname AS Instructor_fname,
i.lname AS Instructor_lname,
i.qualification ,
c.Course_name,
c.Course_date,
p.fname AS Participant_fname,
p.lname AS Participant_lname
FROM
Course c
JOIN
Registration r ON c.Course_ID = r.Course_ID
JOIN
Participant p ON r.Participant_ID = p.Participant_ID
JOIN
Instructor i ON c.Instructor_ID = i.Instructor_ID
WHERE
c.Instructor_ID = 114137866
ORDER BY
c.Course_name,
c.Course_date,
p.lname,
p.fname;
```

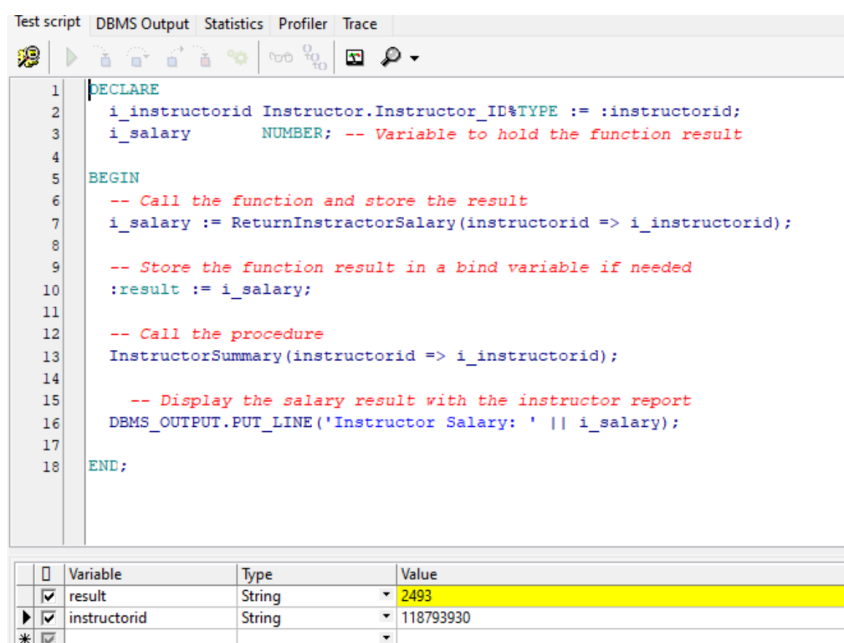
	INSTRUCTOR_FNAME	INSTRUCTOR_LNAME	QUALIFICATION	COURSE_NAME	COURSE_DATE	PARTICIPANT_FNAME	PARTICIPANT_LNAME
1	Brittany	Loggins	nurse excellent	First Aid for Hypothermia	15/08/2024	Glyn	Croucher
2	Brittany	Loggins	nurse excellent	First Aid for Hypothermia	15/08/2024	Rich	Fettis
3	Brittany	Loggins	nurse excellent	First Aid for Hypothermia	15/08/2024	Dun	Tonna

תוכניות ראשיות

תכנית 1:

בתכנית נקבל קלט מהמשתמש של מזהה מדריך. נרצה לקרוא לפונקציה שמחשבת את המשכורת שלו וכן לפרוצדורה שמחזירה דו"ח הכולל את הקורסים שמלמד והמשתתפים בהם. יחד עם דו"ח זה תודפס גם המשכורת.

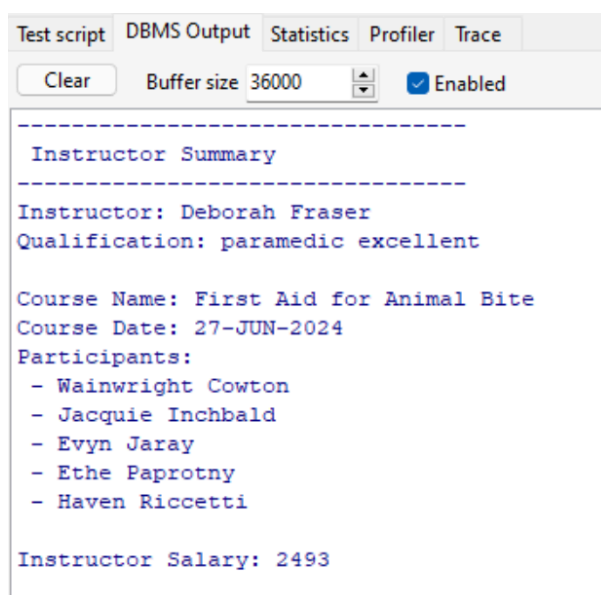
הרצה:



The screenshot shows a SQL IDE with a PL/SQL script in the 'Test script' tab. The script declares a variable `i_instructorid` of type `Instructor.Instructor_ID%TYPE` and `i_salary` of type `NUMBER`. It then calls the function `ReturnInstructorSalary` with `i_instructorid` as an argument, stores the result in `i_salary`, calls the procedure `InstructorSummary` with `i_instructorid`, and finally displays the salary result using `DBMS_OUTPUT.PUT_LINE`. The script ends with `END;`.

Below the script, the 'DBMS Output' tab shows the execution results. The output is as follows:

Variable	Type	Value
result	String	2493
instructorid	String	118793930



The screenshot shows the 'DBMS Output' window with the following output:

```
-----
Instructor Summary
-----
Instructor: Deborah Fraser
Qualification: paramedic excellent

Course Name: First Aid for Animal Bite
Course Date: 27-JUN-2024
Participants:
- Wainwright Cowton
- Jacquie Inchbald
- Evyn Jaray
- Ethe Paprotny
- Haven Riccetti


Instructor Salary: 2493
```

תכנית 2:

בתכנית נרצה לקרוא לפונקציה שמחשבת את ערך המחסן וכן לפרוצדורה שמעבירה משתתפים מקורס אחד לשני.

כמו כן הפונקציה תדפיס את התוצאה של חישוב ערך המחסן.

Test scriptDBMS OutputStatisticsProfilerTrace



```
1 declare
2   v_success      BOOLEAN := TRUE; -- Flag to check if the procedure worked
3   v_function_res NUMBER; -- Variable to hold the function result
4
5 begin
6
7   v_function_res := calculate_location_value(p_category_location => :p_category_location);
8   :result := v_function_res; -- Store the function result in a bind variable
9
10  -- Output the function result
11  DBMS_OUTPUT.PUT_LINE('Function Result: ' || v_function_res);
12
13
14
15  transferParticipants(from_course_id => :from_course_id,
16                     to_course_id => :to_course_id);
17
18 end;
```

	Variable	Type	Value
<input checked="" type="checkbox"/>	from_course_id	String	1018
<input checked="" type="checkbox"/>	to_course_id	String	1356
<input type="checkbox"/>	ref_cur	String	
<input checked="" type="checkbox"/>	p_category_location	String	Storage Room 1
<input checked="" type="checkbox"/>	result	String	6421.994884910485933503836317135549872123
<input checked="" type="checkbox"/>	*		

Test scriptDBMS OutputStatisticsProfilerTrace

Clear

Buffer size10000

☒ Enabled

Function Result: 6421.994884910485933503836317135549872123
Participants transferred successfully