*שלב 3*

(תיעוד הפונקיציות מובנה בהערות)

create or replace package Package is

*-- Author : YURISHO*

*-- Created : 09/06/2015 23:26:18*

*-- Purpose :*

*-- Public function and procedure declarations*

function BooksByPublisherStudy(publisher\_in in varchar2) return number;

function SetWomanSectionCapacity(id in number) return number;

procedure ReportRavClasses(id in number);

procedure ReportWhenVolForRav(id in number);

end Package;

create or replace package body Package is

*--Input: publisher name*

*--Output: What percentage of books being studied are this publisher's books?*

function BooksByPublisherStudy(publisher\_in in varchar2) return number is

cursor cBook is *--books by publisher*

SELECT bookID AS ID FROM book WHERE publisher = publisher\_in;

total\_percentage number := 0;

total\_students integer;

rec\_percentage number;

begin

SELECT COUNT(\*) INTO total\_students FROM Student;

for book\_rec in cBook loop

WITH StudiesTeaches AS

(SELECT DISTINCT Studies.bookID AS bookID

FROM Teaches

INNER JOIN Studies

ON Teaches.bookID = Studies.Bookid)

SELECT SUM(CASE

WHEN bookID = book\_rec.ID THEN

1

ELSE

0

END) / COUNT(\*) \* 100 *--get percentage studying each book by the publisher*

INTO rec\_percentage

FROM StudiesTeaches;

total\_percentage := total\_percentage + rec\_percentage; *--and total the precentage*

end loop;

return total\_percentage;

end;

*--Input: beit midrash id*

*--Output: calculates what woman section capacity should be, updates the value, and returns it.*

*-- if womenSection is 0 (false) then returns -1.*

function SetWomanSectionCapacity(id in number) return number is

WOMAN\_SECTION\_PERCENT CONSTANT number := 0.2;

capacity number;

womenSection char(1);

womenCapacity number;

begin

SELECT capacity, womenSection

INTO capacity, womenSection

FROM BeitMidrashHall

Where bmhID = id;

if womenSection = 1 then

womenCapacity := capacity \* WOMAN\_SECTION\_PERCENT;

UPDATE BeitMidrashHall

SET Womensectioncapacity = womenCapacity

WHERE bmhID = id;

return womenCapacity;

else

return - 1;

end if;

end;

*--Input: ravID*

*--Prints: \* list of beit midrash halls rav teaches in.*

*-- \* list of classrooms rav teaches in.*

*-- \* Calculates salary on a per student basis.*

procedure ReportRavClasses(id in number) is

SALARY\_BMH\_COEFICIANT CONSTANT number := 140.6; *--salary per beit midrash*

SALARY\_CLASS\_COEFICIANT CONSTANT number := 20.7; *--salary per classroom*

SALARY\_BASE CONSTANT number := 400; *--base sallary*

cursor cbmhName is

SELECT DISTINCT BeitMidrashHall.name AS bmhName

FROM Student

INNER JOIN BeitMidrashHall

ON Student.bmhID = BeitMidrashHall.bmhID

WHERE Student.ravID = id *--where rav's students study*

UNION

SELECT BeitMidrashHall.name AS bmhName

FROM BeitMidrashHall

WHERE ravID = id; *-- where rav is head of beit midrash*

cursor cstudentClass is

SELECT DISTINCT class FROM Student WHERE ravID = id;

total\_salary number;

o\_number number; *--temp for outputing numbers*

o\_string varchar2(38); *--temp for outputing strings*

begin

total\_salary := SALARY\_BASE;

SELECT ravName INTO o\_string FROM Rav WHERE Rav.ravID = id;

dbms\_output.put\_line('Rav ID: ' || id || ' Rav Name: ' || o\_string);

dbms\_output.put\_line('-----');

dbms\_output.put\_line('List of beit midrash halls rav teaches in:');

for rec in cbmhName loop

dbms\_output.put\_line(rec.bmhName);

total\_salary := total\_salary + SALARY\_BMH\_COEFICIANT;

end loop;

dbms\_output.put\_line('-----');

dbms\_output.put\_line('list of classrooms rav teaches in:');

for rec in cstudentClass loop

dbms\_output.put\_line(rec.class);

total\_salary := total\_salary + SALARY\_CLASS\_COEFICIANT;

end loop;

dbms\_output.put\_line('-----');

SELECT bankAccountNum INTO o\_number FROM Rav WHERE Rav.ravID = id;

dbms\_output.put\_line('Salary: ' || total\_salary ||

' Bank Account Number: ' || o\_number);

end;

*--Input: ravID*

*--Prints: \* times his students are in volenteer, at what orgenization and address, with details*

procedure ReportWhenVolForRav(id in number) is

cursor cstudentVolTime is

SELECT Name,

organization,

dayOfWeek,

TimeOfDay,

VolunteerGroup.address,

details

FROM VolunteerGroup

INNER JOIN Student

ON Student.vgID = VolunteerGroup.vgID

WHERE ravID = id;

begin

for rec in cstudentVolTime loop

dbms\_output.put\_line('Student: ' || rec.name || ' Volunteer in: ' ||

rec.organization || ' in day: ' ||

rec.dayOfWeek || ' at time: ' || rec.timeOfDay ||

' at address: ' || rec.address);

dbms\_output.put\_line('details: ' || rec.details);

dbms\_output.put\_line('-----');

end loop;

end;

begin

null;

end Package;

**בדיקות:**

*-- Created on 10/06/2015 by YURISHO*

declare

*-- Local variables here*

percent\_count number := 0;

cursor cPublisher is

SELECT DISTINCT publisher FROM book;

begin

*-- Test statements here*

for publisher\_rec in cPublisher loop

percent\_count := percent\_count + Package.BooksByPublisherStudy(publisher\_rec.publisher);

dbms\_output.put\_line('Persent of students who study or being taught from publisher ' ||

publisher\_rec.publisher || ' ' ||

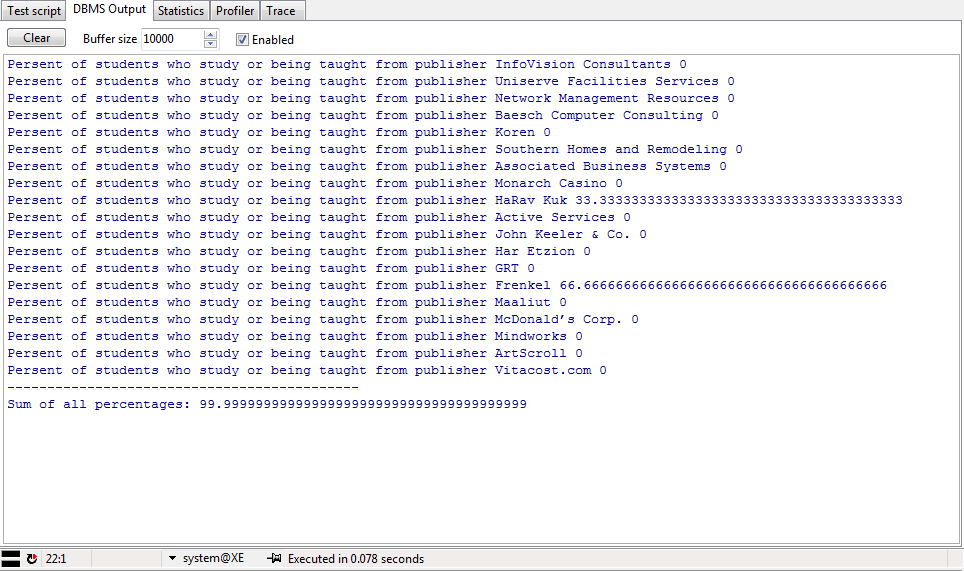
Package.BooksByPublisherStudy(publisher\_rec.publisher));

end loop;

dbms\_output.put\_line('--------------------------------------------');

dbms\_output.put\_line('Sum of all percentages: ' || percent\_count);

end;



*-- Created on 23/06/2015 by YURISHO*

declare

*-- Local variables here*

i integer;

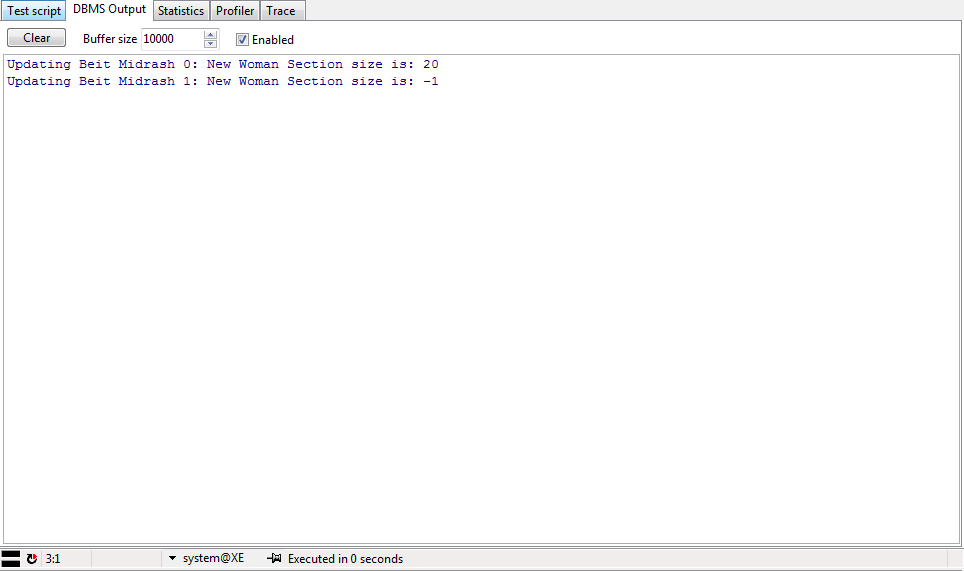
begin

*-- Test statements here*

dbms\_output.put\_line('Updating Beit Midrash 0: New Woman Section size is: ' ||Package.SetWomanSectionCapacity(0));

dbms\_output.put\_line('Updating Beit Midrash 1: New Woman Section size is: ' ||Package.SetWomanSectionCapacity(1));

end;



*-- Created on 10/06/2015 by YURISHO*

declare

*-- Local variables here*

cursor cRav is

SELECT ravID FROM Rav;

begin

*-- Test statements here*

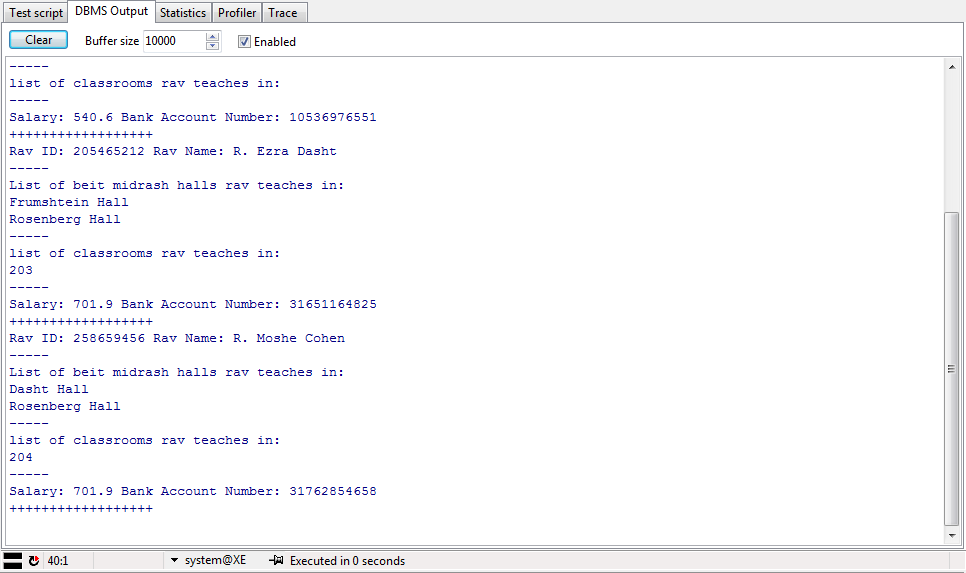
for rav\_id in cRav loop

Package.ReportRavClasses(rav\_id.ravID);

dbms\_output.put\_line('++++++++++++++++++');

end loop;

end;



*-- Created on 23/06/2015 by YURISHO*

declare

*-- Local variables here*

cursor cRav is

SELECT ravID FROM Rav;

begin

*-- Test statements here*

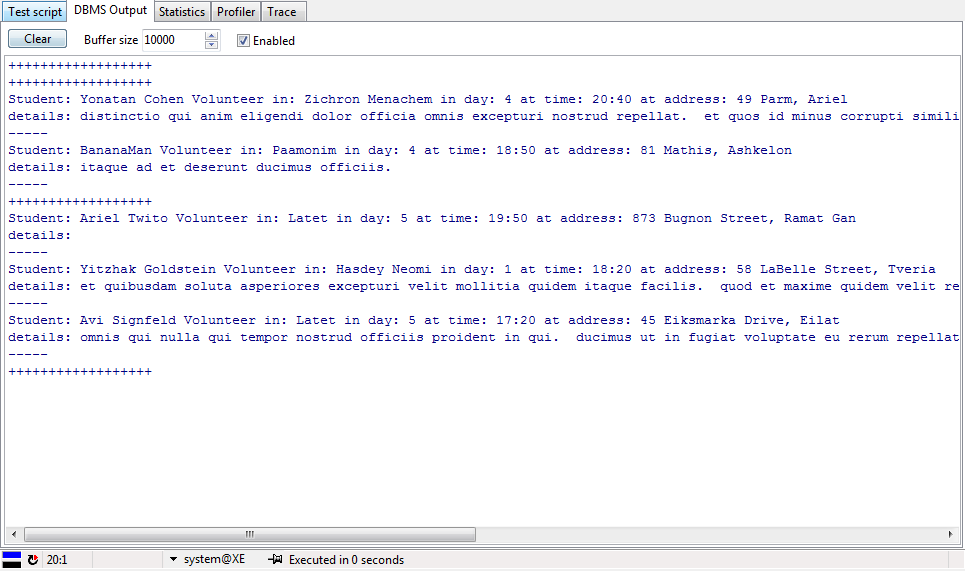
for rav\_id in cRav loop

Package.ReportWhenVolForRav(rav\_id.ravID);

dbms\_output.put\_line('++++++++++++++++++');

end loop;

end;



***Triggers:***

תעד כל מחיקה מהטבלה Studies – רשום את זמן המחיקה, שם המשתמש המוחק והקשר בין איזה תלמיד וספר נמחק.

create or replace trigger RecordStudiesDelete

before delete on Studies

for each row

when (old.stdID > 0)

declare

*-- local variables here*

name\_student varchar2(38);

name\_book varchar2(38);

begin

SELECT Student.name

INTO name\_student

FROM Student

WHERE Student.StdID = :old.stdID;

SELECT Book.name

INTO name\_book

FROM Book

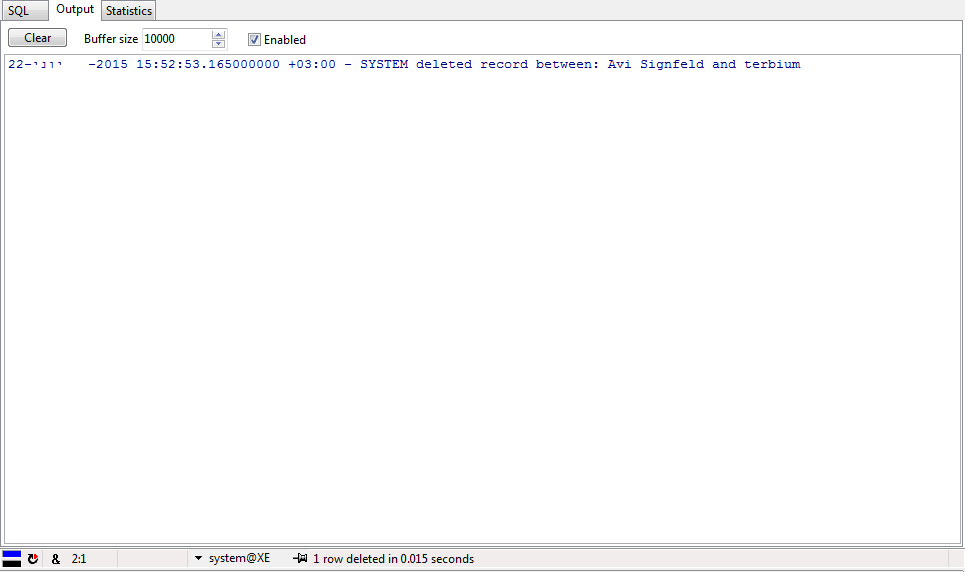
WHERE Book.bookID = :old.bookID;

dbms\_output.put\_line(SYSTIMESTAMP || ' - ' || USER ||

' deleted record between: ' || name\_student ||

' and ' || name\_book);

end RecordStudiesDelete;



בכל שינוי לבית המדרש, אם קיימת עזרת נשים, תעד כמה מקומות יש בעזרת גברים.

create or replace trigger CalcNonWomanSectionLeft

after update on beitmidrashhall

for each row

declare

manSectionCapacity number;

begin

if :new.womensection = 1 then

manSectionCapacity := :new.capacity - :new.womensectioncapacity;

dbms\_output.put\_line(:new.name || ' - man section capacity left: ' ||

manSectionCapacity);

end if;

end CalcNonWomanSectionLeft;

