שלב 4:

מטלה 9:

1. פרוצדורות:
   1. הדפסת הסוכן /ים שסגר הכי הרבה עסקאות:

הפרוצדורה מדפיסה את הסוכן הכי טוב שביצע הכי הרבה עסקאות רכב או שביצע הכי הרבה השאלות ספרים. הפרוצדורה לא לוקחת בחשבון סוכן שביצע גם השכרות רכב וגם השאלות ספרים בחישוב הסוכן המוצלח.

הקוד:

create or replace procedure best\_agent is

cursor bext\_agents is(

select agent\_id from agent where agent\_id in(

select agent\_id from (

select agent\_id, count(\*) as deels

from rent natural join agent

group by agent\_id)

where deels =(select max (num\_deels)

from( select agent\_id, count(\*) as num\_deels

from rent natural join agent

group by agent\_id))

UNION

select agent\_id from (

select agent\_id, count(\*) as hashalot

from hashala natural join agent

group by agent\_id)

where hashalot =(select max (num\_hashalot)

from( select agent\_id, count(\*) as num\_hashalot

from hashala natural join agent

group by agent\_id))));

begin

dbms\_output.put\_line('The best agents: ');

FOR agent IN bext\_agents

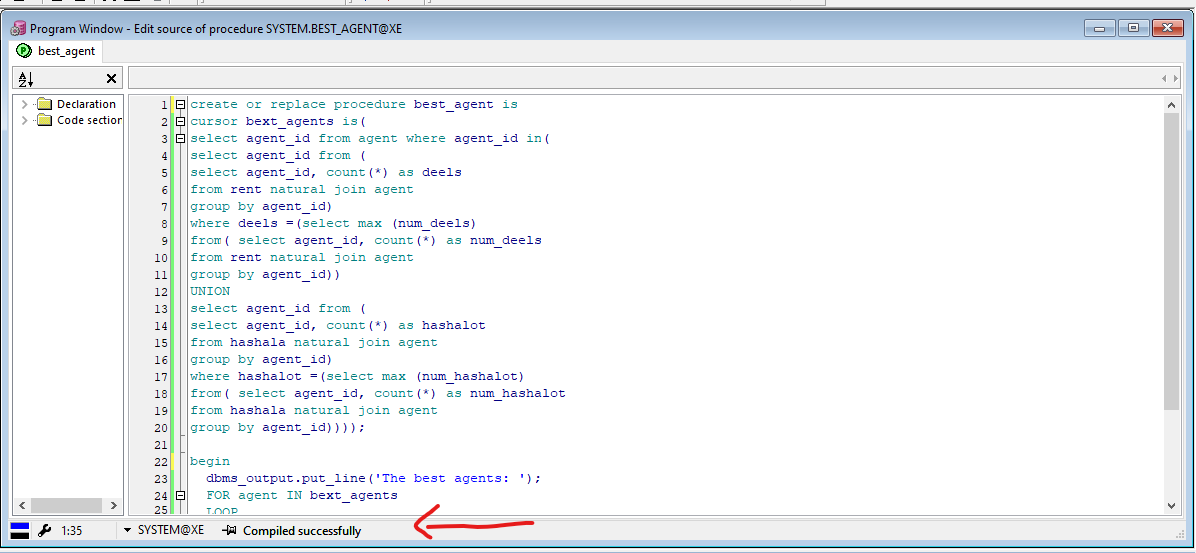
LOOP

dbms\_output.put\_line('agent id: '|| agent.agent\_id);

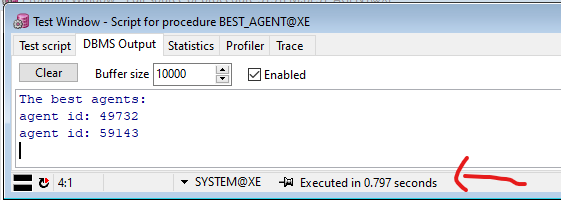
END LOOP;

end;

הקוד עובר קומפילציה:



הפלט:



* 1. הפרוצדורה מקבלת כקלט author\_id ומדפיסה את כל הספרים שהוא חיבר.

הקוד:

create or replace procedure books\_and\_authors(author in out varchar) is

cursor books is select \* from book where author\_id = author;

begin

dbms\_output.put\_line('The author is: ' || author);

dbms\_output.put\_line('The authors books are: ');

FOR book IN books

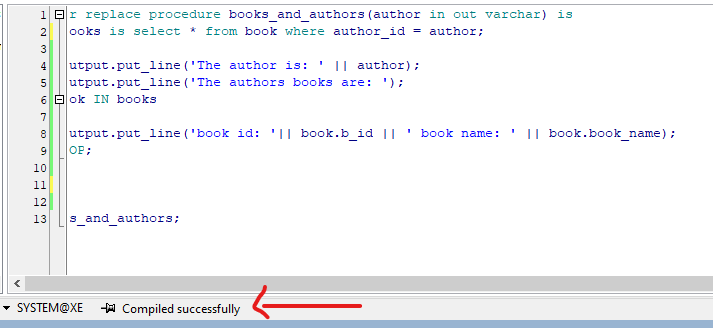
LOOP

dbms\_output.put\_line('book id: '|| book.b\_id || ' book name: ' || book.book\_name);

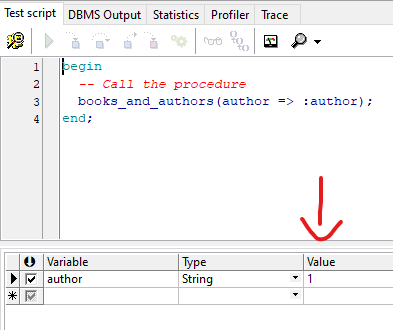
END LOOP;

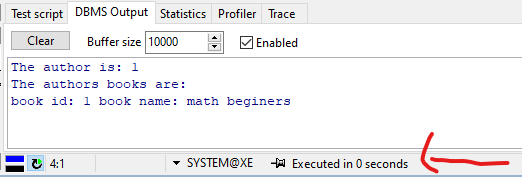
end books\_and\_authors;

הקוד עובר קומפילציה:



הקלט:



הפלט: 

1. פונקציות
   1. הפונקציה מקבלת report\_id, מחזירה את הreport\_isPaid שלו, ומשנה את הreport\_isPaid שלו ל ‘YES’.

הקוד:

create or replace function update\_report(report in NUMBER) return CHAR is

isPaid CHAR;

begin

select report\_isPaid into isPaid from report where report\_id = report;

if sql%notfound then

dbms\_output.put\_line('report not exist');

elsif sql%found then

update report

SET report\_isPaid = 'YES'

where report\_id = report;

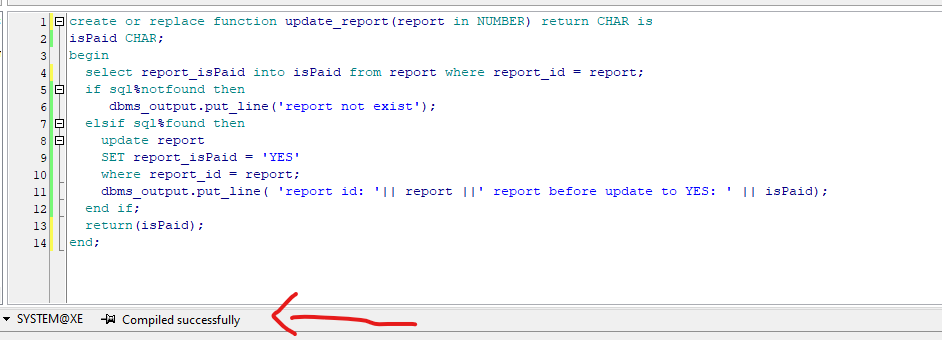
dbms\_output.put\_line( 'report id: '|| report ||' report before update to YES: ' || isPaid);

end if;

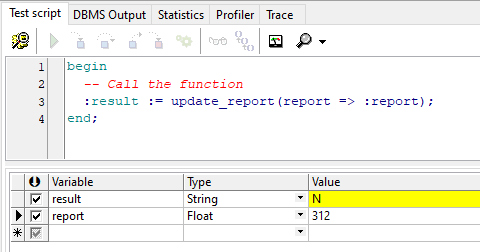
return(isPaid);

end;

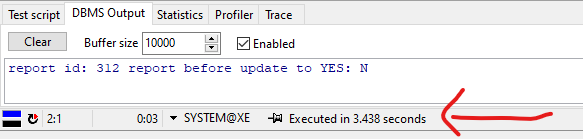
הקוד עובר קומפילציה:



הקלט והreturn value של התוכנית:



ההדפסות וזמן הריצה:



2.2 הפונקציה מקבלת vehicle\_id והנחה ומחזירה את המחיר הרכב לאחר ההנחה.

הקוד:

create or replace function calculates\_discount(licensenumber in number, discount in number) return number is

price number;

vehicle\_model model.model\_id%type;

begin

select model\_id into vehicle\_model from vehicle where vehicle\_licensenumber = licensenumber;

update model

SET model\_price = model\_price - (model\_price \* discount / 100)

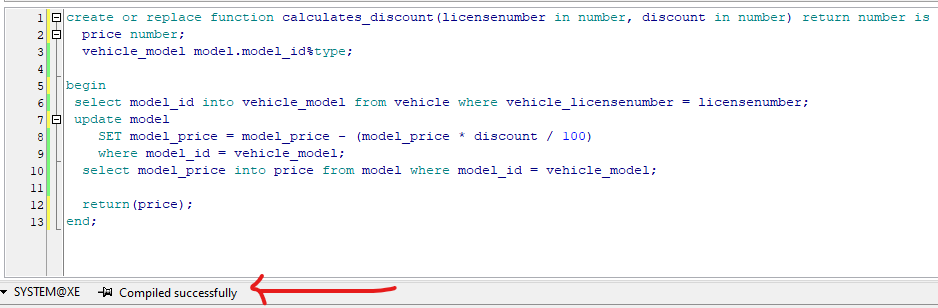
where model\_id = vehicle\_model;

select model\_price into price from model where model\_id = vehicle\_model;

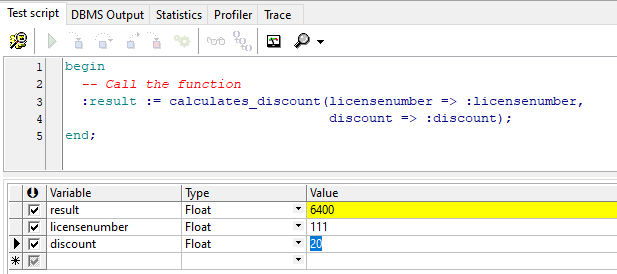
return(price);

end;

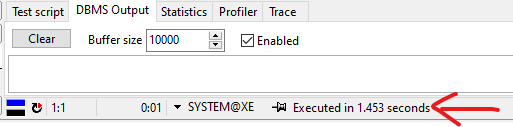
הקוד עובר קומפילציה:



הקלט והreturn value:

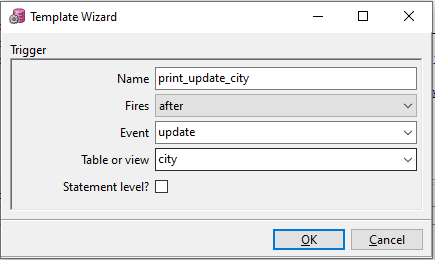


זמן הריצה:



1. טריגרים:
   1. טריגר שמדפיס שהיה עדכון בכל פעם שמעדכנים את הareaid בטבלה city.

יצירת הטריגר:



הקוד:

create or replace trigger print\_update\_city

after update of areaid on city

for each row

declare

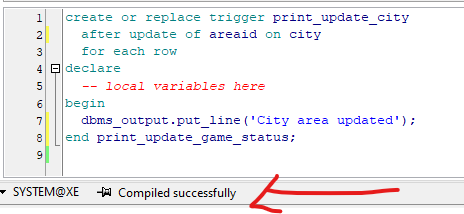
*-- local variables here*

begin

dbms\_output.put\_line('City area updated');

end print\_update\_game\_status;

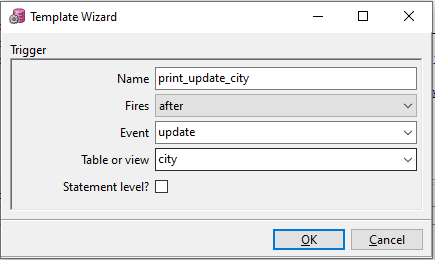
הקוד עובר קומפילציה:



...

* 1. טריגר שמדפיס בכל פעם שסוכן נוסף בהצלחה לטבלת agent.

יצירת הטריגר:



הקוד:

create or replace trigger insert\_agent

after insert on agent

for each row

declare

begin

dbms\_output.put\_line(:new.agent\_id || ' successfully added');

end insert\_agent;

הקוד עובר קומפילציה:

