

Ex.No.: 12

Date:

05/11/25

WORKING WITH CURSOR, PROCEDURES AND FUNCTIONS

AIM:

Create PL/SQL Blocks to perform the Item Transaction Operations using CURSOR, FUNCTION and PROCEDURE.

ALGORITHM:

STEP-1: Start.

STEP-2: Create two tables Item Master and Item Trans.

itemmaster(itemid , itemname, stockonhand)

itemtrans(itemid ,itemname ,dateofpurchase ,quantity)

STEP-3: Create a PROCEDURE with id, name and quantity as parameters which make a call to the FUNCTION by passing id, name, dop. and quantity as parameters dop is set as sysdate.

STEP-4: Using FUNCTION fetch each record from the table Item Master using CURSOR inside a Loop statement,

If Item Master's ItemId is equal to the entered ID value then exit the loop otherwise fetch the next record.

loop

 fetch master into masterrec

 exit when master%notfound

 if masterrec.itemid=id then

 exit;

 end if;

end loop;

STEP-5: If Itemmaster's itemid = id then,

 Add the Itemmaster's stockonhand with the given quantity and update the ItemMaster table and insert the Item information into the ItemTrans table.

STEP-6: Else, if the inputed item is not present in the ItemMaster table then insert the

...the table, the chair, the floor, the window, the door, the lamp, the bed, the table, the chair, the floor, the window, the door, the lamp, the bed,

the table, the chair, the floor, the window, the door, the lamp, the bed,

the table, the chair, the floor, the window, the door, the lamp, the bed,

CREATING THE TABLE, SITTING AND DISPLAYING THE CONTENTS

the table, the chair, the floor, the window, the door, the lamp, the bed, the table, the chair, the floor, the window, the door, the lamp, the bed,

the table, the chair, the floor, the window, the door, the lamp, the bed,

```
SQL> insert into ititems values(102, 3000, 1600, 202);
1 row created.
```

```
SQL> insert into ititems values(103, 4000, 600, 202);
1 row created.
```

```
SQL> select * from ititems;
```

ITEMID	ACTUALPRICE	ORDID	PRODID
101	2000	500	201
102	3000	1600	202
103	4000	600	202

PROGRAM FOR GENERAL PROCEDURE – SELECTED RECORD'S PRICE IS INCREMENTED BY 500 , EXECUTING THE PROCEDURE CREATED AND DISPLAYING THE UPDATED TABLE

```
SQL> create procedure itsum(identity number, total number) is price number;
2 null_price exception;
3 begin
4 select actualprice into price from ititems where itemid=identity;
5 if price is null then
6 raise null_price;
7 else
8 update ititems set actualprice=actualprice+total where itemid=identity;
9 end if;
10 exception
11 when null_price then
12 dbms_output.put_line('price is null');
13 end;
14 /
Procedure created.
```

```
SQL> exec itsum(101, 500);
PL/SQL procedure successfully completed.
```

```
SQL> select * from ititems;
```

ITEMID	ACTUALPRICE	ORDID	PRODID
101	2500	500	201
102	3000	1600	202
103	4000	600	202

PROCEDURE FOR 'IN' PARAMETER – CREATION, EXECUTION

```
SQL> set serveroutput on;
```

WILHELMUS SEPPÄNEN: CREATION EXECUTION

112 L'ESPRESSO - INTELLIGENTESIA - CULTURE ET POLITIQUE

```
4 end;
5 /
Procedure created.

SQL> declare
2 a number:=7;
3 begin
4 itit(a);
5 dbms_output.put_line('The updated value is '||a);
6 end;
7 /
The updated value is 8
PL/SQL procedure successfully completed.
```

CREATE THE TABLE 'ITTRAIN' TO BE USED FOR FUNCTIONS

```
SQL>create table ittrain ( tno number(10), tfare number(10));
Table created.
```

```
SQL>insert into ittrain values (1001, 550);
1 row created.
```

```
SQL>insert into ittrain values (1002, 600);
1 row created.
```

```
SQL>select * from ittrain;
TNO      TFARE
-----
1001      550
1002      600
```

PROGRAM FOR FUNCTION AND IT'S EXECUTION

```
SQL> create function aaa (trainnumber number) return number is
2 trainfunction ittrain.tfare % type;
3 begin
4 select tfare into trainfunction from ittrain where tno=trainnumber;
5 return(trainfunction);
6 end;
7 /
```

Function created.

```
SQL> set serveroutput on;
```

```
SQL> declare
2 total number;
3 begin
4 total:=aaa (1001);
5 dbms_output.put_line('Train fare is Rs. '|total);
6 end;
7 /
```

Train fare is Rs.550

PL/SQL procedure successfully completed.

FACTORIAL OF A NUMBER USING FUNCTION — PROGRAM AND EXECUTION

```
SQL> create function ifact (a number) return number is
2 fact number:=1;
3 b number;
4 begin
5 b:=a;
6 while b>0
7 loop
8 fact:=fact*b;
9 b:=b-1;
10 end loop;
11 return(fact);
12 end;
13 /
```

Function created.

```
SQL> set serveroutput on;
```

```
SQL> declare
2 a number:=7;
3 f number(10);
4 begin
5 f:=ifact(a);
6 dbms_output.put_line('The factorial of the given number is'||f);
7 end;
8 /
```

The factorial of the given number is 5040

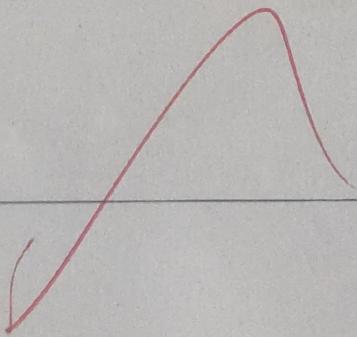
PL/SQL procedure successfully completed.

Program 1

FACTORIAL OF A NUMBER USING FUNCTION

CREATE OR REPLACE FUNCTION factorial (n IN NUMBER)
RETURN NUMBER IS

```
result NUMBER := 1;  
i NUMBER;  
  
BEGIN  
    IF n < 0 THEN  
        RETURN -1;  
    ELSIF n = 0 OR n = 1 THEN  
        RETURN 1;  
    ELSE  
        FOR i IN 2 ... n LOOP  
            result := result * i;  
        END LOOP;  
        RETURN result;  
    END IF;  
END;  
/  
BEGIN      -- TO use the function  
DBMS_OUTPUT.PUT_LINE('Factorial of 5: ' || factorial(5));  
DBMS_OUTPUT.PUT_LINE('Factorial of 0: ' || factorial(0));  
END;  
/
```



Program:

Write a PL/SQL program using Procedures IN,INOUT,OUT parameters to retrieve the corresponding book information in library

```
CREATE TABLE Books (book_id NUMBER PRIMARY KEY,  
title VARCHAR2(100),  
author VARCHAR2(100),  
copies NUMBER  
);  
INSERT INTO books VALUES (1, 'Clean', 'Robert', 5);  
COMMIT;  
CREATE OR REPLACE PROCEDURE get_book_info(  
    p_book_id IN books.book_id%TYPE,  
    p_title OUT books.title%TYPE,  
    p_copies IN OUT books.copies%TYPE) IS  
BEGIN  
    SELECT title, copies INTO p_title, p_copies FROM books;  
EXCEPTION  
    WHEN NO_DATA_FOUND THEN p_title := NULL;  
    p_copies := NULL;  
END;
```

TO WRITE A PL/SQL BLOCK TO DISPLAY THE EMPLOYEE ID AND EMPLOYEE NAME WHERE DEPARTMENT NUMBER IS 11 USING EXPLICIT CURSORS

```
1 declare  
2 cursor cemp is select eid,sal from ssempp where dno=11;  
3 ecode ssempp.eid%TYPE;  
4 esal ssempp.sal%TYPE;  
5 begin  
6 open cemp;  
7 loop  
8 fetch cemp into ecode,esal;  
9 exit when cemp%notfound;  
10 dbms_output.put_line('Employee code and employee salary are' || ecode 'and' || esal);
```

```
11 end loop;
12 close cenl;
13* end;
```

SQL> /

Employee code and employee salary are 1 and 39000
Employee code and employee salary are 5 and 35000
Employee code and employee salary are 6 and 23000

PL/SQL procedure successfully completed.

TO WRITE A PL/SQL BLOCK TO UPDATE THE SALARY BY 5000 WHERE THE JOB IS LECTURER , TO CHECK IF UPDATES ARE MADE USING IMPLICIT CURSORS AND TO DISPLAY THE UPDATED TABLE

```
SQL> declare
2  county number;
3 begin
4 update ssempp set sal=sal+10000 where job='lecturer';
5 county:= sql%rowcount;
6 if county > 0 then
7 dbms_output.put_line('The number of rows are'|| county);
8 end if;
9 if sql %found then
10 dbms_output.put_line('Employee record modification successful');
11 else if sql%notfound then
12 dbms_output.put_line('Employee record is not found');
13 end if;
14 end if;
15 end;
16 /
```

The number of rows are 3

Employee record modification successful

PL/SQL procedure successfully completed.

SQL> select * from ssempp;

EID	ENAME	JOB	SAL	DNO
1	nala	lecturer	44000	11
2	kala	seniorlecturer	20000	12
5	ajay	lecturer	40000	11
6	vijay	lecturer	28000	11
3	nila	professor	60000	12

Evaluation Procedure	Marks awarded
PL/SQL Procedure(5)	
Program/Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	

ESULT :

thus all the above PL/SQL statements were executed.

