**********Experiment no:-06**********

Author:Saurabh Khandagale
Roll No:46
Date :02- October -2020

EXPERIMENT:-06

AIM:-

To Write and Execute Stored Procedure and Stored Function using Oracle 11g.

Problem Statement:

Using the relation schemata established in Experiments - 02, 03, and 05, create and
execute the mentioned stored functions and stored procedures. QUERIES:

==========Query-01==========

Write SQL code to compile and execute a stored procedure - SHOW_EMPLOYEE,to list employee details for the input variable ENO holding employee number. (Use EMPP Table).

==========

```
CREATE OR REPLACE PROCEDURE SHOW_EMPLOYEE(NO IN NUMBER )
IS
EE_NO NUMBER(10);
EN VARCHAR(30);
HDATE DATE;
DESG VARCHAR(30);
SAL NUMBER(20);
BEGIN
SELECT EID,ENAME,HIREDATE,DESIGNATION,SALARY INTO EE_NO,EN,HDATE,DESG,SAL
FROM EMPP  WHERE EID=NO;
DBMS_OUTPUT.PUT_LINE('OUTPUT');
DBMS_OUTPUT.PUT_LINE('-----------------------------------------------------');

DBMS_OUTPUT.PUT_LINE(EE_NO||' '||EN||' '||HDATE||' '||DESG||' '||SAL);

DBMS_OUTPUT.PUT_LINE('-----------------------------------------------------');

END  SHOW_EMPLOYEE;
/


DECLARE

NO NUMBER;
BEGIN
DBMS_OUTPUT.PUT_LINE('ENTER ENO OF EMPLOYEE');
NO:='&NO';
 SHOW_EMPLOYEE(NO);
END;
/
```

```
SQL> /
Enter value for no: 7111
old   6: NO:='&NO';
new   6: NO:='7111';
ENTER ENO OF EMPLOYEE
OUTPUT
-----------------------------------------------------
7111 Albert Greenfield 12-JUL-16 Research Asst. 48200
-----------------------------------------------------

PL/SQL procedure successfully completed.



SQL> SELECT OBJECT_NAME,OBJECT_ID DATA,OBJECT_TYPE,STATUS
FROM USER_OBJECTS WHERE OBJECT_NAME='SHOW_EMPLOYEE';

OBJECT_NAME           DATA OBJECT_TYPE         STATUS
--------------- ---------- ------------------ -------
SHOW_EMPLOYEE        22504 PROCEDURE          VALID
```

```
      ================================Query02=================================
      Write SQL code to compile and execute a stored procedure - ADD_EMPLOYEE,
      to add a record to EMPP table. Check the existence of the created
      procedure using USER_OBJECTS view. Use this procedure to insert
      following records
      ========================================================================
```

```
CREATE  OR REPLACE PROCEDURE ADD_EMPLOYEE(NO EMPP.EID%TYPE,EN
EMPP.ENAME%TYPE,HDATE EMPP.HIREDATE%TYPE,DESIG EMPP.DESIGNATION%TYPE,SAL
EMPP.SALARY%TYPE )
AS
NO1 NUMBER;
EN1 VARCHAR(20);
HDATE1 DATE;
DESIG1 VARCHAR(20);
SAL1 NUMBER(20);
CHK VARCHAR(20);
NN VARCHAR(30);
BEGIN

Select OBJECT_NAME INTO NN FROM USER_OBJECTS WHERE OBJECT_NAME='ADD_EMPLOYEE';
DBMS_OUTPUT.PUT_LINE('CREATED PROCEDURE '||NN);
INSERT INTO EMPP VALUES(NO,EN,HDATE,DESIG,SAL);


END  ADD_EMPLOYEE;
/
```

```
DECLARE

NO NUMBER;
EN VARCHAR(20);
HDATE DATE;
DESIG VARCHAR(20);
SAL NUMBER(20);
BEGIN
DBMS_OUTPUT.PUT_LINE('ENTER EMPLOYEE DETAILS');
NO:='&NO';
EN:='&EN';
HDATE:='&HDATE';
DESIG:='&DESIG';
SAL:='&SAL';
ADD_EMPLOYEE(NO,EN,HDATE,DESIG,SAL);
DBMS_OUTPUT.PUT_LINE('---Record Inserted---');

END;
/
Enter value for no: 7118
old  10: NO:='&NO';
new  10: NO:='7118';
Enter value for en: Saurabh Khandagale
old  11: EN:='&EN';
new  11: EN:='Saurabh Khandagale';
Enter value for hdate: 07-jul-2020
old  12: HDATE:='&HDATE';
new  12: HDATE:='07-jul-2020';
Enter value for desig: Teaching Asst.
old  13: DESIG:='&DESIG';
new  13: DESIG:='Teaching Asst.';
Enter value for sal: 25000
old  14: SAL:='&SAL';
new  14: SAL:='25000';
CREATED PROCEDURE ADD_EMPLOYEE
---Record Inserted---

PL/SQL procedure successfully completed.

SQL> /
Enter value for no: 7119
old  10: NO:='&NO';
new  10: NO:='7119';
Enter value for en: Atulya Bharat
old  11: EN:='&EN';
new  11: EN:='Atulya Bharat';
Enter value for hdate: 03-Aug-2005
old  12: HDATE:='&HDATE';
new  12: HDATE:='03-Aug-2005';
Enter value for desig: Professor
old  13: DESIG:='&DESIG';
new  13: DESIG:='Professor';
Enter value for sal: 162000
old  14: SAL:='&SAL';
new  14: SAL:='162000';
CREATED PROCEDURE ADD_EMPLOYEE
---Record Inserted---

PL/SQL procedure successfully completed.
```

```
SQL> commit;

Commit complete.

SQL> select * from empp;

       EID ENAME                HIREDATE  DESIGNATION         SALARY
---------- -------------------- --------- -------------- ----------
      7102 Samantha Jones       08-NOV-06 Professor          146500
      7101 Eugene Sabatini      10-OCT-06 Professor          150000
      7103 Alexander Lloyd      01-FEB-07 Professor          148000
      7104 Simon Downing        01-SEP-07 Professor          138400
      7107 Christov Plutnik     01-SEP-08 Asso. Professor    127400
      7105 Christina Mulboro    15-JUL-08 Asso. Professor    127400
      7106 Dolly Silverline     17-AUG-08 Asso. Professor    127400
      7108 Ellena Sanchez       12-NOV-09 Asso. Professor    119700
      7109 Martina Jacobson     15-NOV-09 Asst. Professor     91000
      7110 William Smithfield   23-JUN-10 Asst. Professor     86400
      7111 Albert Greenfield    12-JUL-16 Research Asst.      48200
      7112 James Washington     22-AUG-17 Research Asst.      44600
      7113 Julia Martin         01-DEC-18 Teaching Asst.      35600
      7114 Larry Gomes          18-MAY-19 Teaching Asst.      32850
      7115 Svetlana Sanders     15-JAN-20 Teaching Asst.      30000
      7116 Lovelyn Brendon      17-JUL-20 Teaching Asst.      30000
      7117 Hector Hercules      01-AUG-20 Teaching Asst.      32200
      7118 Saurabh Khandagale   07-JUL-20 Teaching Asst.      25000
      7119 Atulya Bharat        03-AUG-05 Professor          162000

19 rows selected.
```

```
=====================================Query03============================
Write SQL code to compile and execute the stored procedure - REMOVE_EMPLOYEE,
which will remove the employee record(s) from EMPP table when supplied with an
input name phrase (entered always as lower case) indicating employee name (use
EMPP table). If the matching employee is not found, an appropriate exception
should be raised
========================================================================
[Note:-I have use Regex to Match the String]
SQL> select * from empp;

       EID ENAME                HIREDATE  DESIGNATION         SALARY
---------- -------------------- --------- -------------- ----------
      7102 Samantha Jones       08-NOV-06 Professor           146500
      7101 Eugene Sabatini      10-OCT-06 Professor           150000
      7103 Alexander Lloyd      01-FEB-07 Professor           148000
      7104 Simon Downing        01-SEP-07 Professor           138400
      7107 Christov Plutnik     01-SEP-08 Asso. Professor     127400
      7105 Christina Mulboro    15-JUL-08 Asso. Professor     127400
      7106 Dolly Silverline     17-AUG-08 Asso. Professor     127400
      7108 Ellena Sanchez       12-NOV-09 Asso. Professor     119700
      7109 Martina Jacobson     15-NOV-09 Asst. Professor      91000
      7110 William Smithfield   23-JUN-10 Asst. Professor      86400
      7111 Albert Greenfield    12-JUL-16 Research Asst.       48200
      7112 James Washington     22-AUG-17 Research Asst.       44600
      7113 Julia Martin         01-DEC-18 Teaching Asst.       35600
      7114 Larry Gomes          18-MAY-19 Teaching Asst.       32850
      7115 Svetlana Sanders     15-JAN-20 Teaching Asst.       30000
      7116 Lovelyn Brendon      17-JUL-20 Teaching Asst.       30000
      7117 Hector Hercules      01-AUG-20 Teaching Asst.       32200
      7118 Saurabh Khandagale   07-JUL-20 Teaching Asst.       25000
      7119 Atulya Bharat        03-AUG-05 Professor           162000

19 rows selected.

[Demo]
SQL> SELECT INITCAP('saurabh khandagale') from dual;

INITCAP('SAURABHKH
------------------
Saurabh Khandagale


CREATE  OR REPLACE PROCEDURE REMOVE_EMPLOYEE(EN EMPP.ENAME%TYPE)
AS
EN1 EMPP.ENAME%TYPE;
NN EMPP.ENAME%TYPE;
D1 NUMBER(4);
D2 NUMBER(4);
D3 NUMBER(4);

BEGIN
EN1:=EN;
NN:=INITCAP(EN1);
DBMS_OUTPUT.PUT_LINE(NN);
SELECT COUNT(*) INTO D1 FROM EMPP;
DELETE FROM   EMPP  WHERE REGEXP_LIKE(ename,'^'||NN);
SELECT COUNT(*) INTO D2 FROM EMPP;
D3:=D1-D2;
DBMS_OUTPUT.PUT_LINE('ROWS  AFFECTED '||D3);
```

```
END REMOVE_EMPLOYEE;
/

DECLARE
EN EMPP.ENAME%TYPE;
BEGIN
DBMS_OUTPUT.PUT_LINE('ENTER EMPLOYEE DETAILS');
EN:='&EN';
REMOVE_EMPLOYEE(EN);
END;
/
[e_name not present]
Enter value for en: ee
old   5: EN:='&EN';
new   5: EN:='ee';
ENTER EMPLOYEE DETAILS
ROWS  AFFECTED 0


[e_name  present]

Enter value for en: saurabh
old   5: EN:='&EN';
new   5: EN:='saurabh';
ENTER EMPLOYEE DETAILS
Saurabh
ROWS  AFFECTED 1



SQL> select * from empp;

       EID ENAME                HIREDATE  DESIGNATION         SALARY
---------- -------------------- --------- -------------- ----------
      7102 Samantha Jones       08-NOV-06 Professor          146500
      7101 Eugene Sabatini      10-OCT-06 Professor          150000
      7103 Alexander Lloyd      01-FEB-07 Professor          148000
      7104 Simon Downing        01-SEP-07 Professor          138400
      7107 Christov Plutnik     01-SEP-08 Asso. Professor    127400
      7105 Christina Mulboro    15-JUL-08 Asso. Professor    127400
      7106 Dolly Silverline     17-AUG-08 Asso. Professor    127400
      7108 Ellena Sanchez       12-NOV-09 Asso. Professor    119700
      7109 Martina Jacobson     15-NOV-09 Asst. Professor     91000
      7110 William Smithfield   23-JUN-10 Asst. Professor     86400
      7111 Albert Greenfield    12-JUL-16 Research Asst.      48200
      7112 James Washington     22-AUG-17 Research Asst.      44600
      7113 Julia Martin         01-DEC-18 Teaching Asst.      35600
      7114 Larry Gomes          18-MAY-19 Teaching Asst.      32850
      7115 Svetlana Sanders     15-JAN-20 Teaching Asst.      30000
      7116 Lovelyn Brendon      17-JUL-20 Teaching Asst.      30000
      7117 Hector Hercules      01-AUG-20 Teaching Asst.      32200
      7119 Atulya Bharat        03-AUG-05 Professor          162000

18 rows selected.

[Deleted Record Inserted After performing operation].
```

```
=====================================Query04============================
Write SQL code to compile and execute the stored function - CHECK_ITEM that
will report status as 1 if items with mentioned P_CODE are present in the
inventory, otherwise reports status as 0. No exceptions to be handled.
========================================================================

SQL> CREATE TABLE ITEMS  AS (SELECT P_CODE ,DESCRIPT AS DESCR,P_DATE AS
IN_DATE,P_MIN AS MIN_QTY,QTY, P_PRICE AS PRICE,V_CODE FROM PRODUCT);

Table created.

P_COD DESCR              IN_DATE      MIN_QTY        QTY      PRICE     V_CODE
----- ------------------ --------- ---------- ---------- ---------- ----------
AB112 Power Drill        03-NOV-19          5          8     109.99      25595
SB725 7.25in Saw Blade   13-DEC-19         15         32      14.99      21344
SB900 9.00 in Saw Blade  13-NOV-19         12         18      17.49      21344

CL050 Hrd. Spring 1/2in  15-JAN-20          5         23      43.99      23119
SH200 Sledge Hammer      05-JUL-20          3         10       25.8      24992
ZZ999 Cordless Drill     10-JUL-20         40        200       25.5      24992
AB212 Power Drill        03-AUG-20          3         15        275      24992

22 rows selected.

SQL> ALTER TABLE ITEMS ADD PRIMARY KEY (P_CODE);

Table altered.

Commit complete.
SQL> ALTER TABLE ITEMS  MODIFY IN_DATE DEFAULT SYSDATE;

Table altered.

SQL> ALTER TABLE ITEMS  MODIFY MIN_QTY DEFAULT 2 ;

Table altered.
```

```
SQL> SELECT CONSTRAINT_NAME,CONSTRAINT_TYPE,TABLE_NAME, DELETE_RULE
     FROM USER_CONSTRAINTS
     WHERE TABLE_NAME='ITEMS';

CONSTRAINT_NAME                C TABLE_NAME                    DELETE_RU
------------------------------ - ------------------------------ ---------
SYS_C007878                    C ITEMS
SYS_C007879                    C ITEMS
SYS_C007880                    C ITEMS
SYS_C007881                    C ITEMS
SYS_C007882                    C ITEMS
SYS_C007883                    C ITEMS
SYS_C007884                    P ITEMS

7 rows selected.
CREATE OR REPLACE  FUNCTION CHECK_ITEM(CODE ITEMS.P_CODE%TYPE)
RETURN  NUMBER
IS
CD ITEMS.P_CODE%TYPE:=NULL;
STATUS NUMBER;
SS NUMBER;
BEGIN
SELECT COUNT(*) INTO SS   FROM  ITEMS WHERE P_CODE=CODE ;
IF (SS<1)  THEN
RETURN 0;
ELSE
RETURN 1;
END IF;
END  CHECK_ITEM;
/
DECLARE
NO NUMBER(3);
BEGIN
NO:=CHECK_ITEM('AB112');
DBMS_OUTPUT.PUT_LINE(NO);
END;
/

0

PL/SQL procedure successfully completed.


SQL> DECLARE
  2  NO NUMBER(3);
  3  BEGIN
  4  NO:=CHECK_ITEM('AB112');
  5  DBMS_OUTPUT.PUT_LINE(NO);
  6  END;
  7  /
1

PL/SQL procedure successfully completed.
```

==========Query05==========
Write a SQL code to compile and execute the stored procedure - ADD_ITEM, that will insert an item in ITEMS table with given particulars - item code, item description, invoice date, quantity of purchase, minimum quantity, item price and supplier code
==========

```
CREATE OR REPLACE PROCEDURE ADD_ITEM
(P_CODE1 ITEMS.P_CODE%TYPE,
DESCR1 ITEMS.DESCR%TYPE,
IN_DATE1 ITEMS.IN_DATE%TYPE,
QTY1 ITEMS.QTY%TYPE,
MIN_QTY1 ITEMS.MIN_QTY%TYPE,
PRICE ITEMS.PRICE%TYPE
,V_CODE ITEMS.V_CODE%TYPE)
IS
BEGIN
INSERT INTO ITEMS VALUES(P_CODE1,DESCR1,IN_DATE1,QTY1, MIN_QTY1,PRICE,V_CODE);
DBMS_OUTPUT.PUT_LINE('ROW AFFECTED '|| SQL%ROWCOUNT||' #DATA_Affected# ');
END;
/
SQL> BEGIN
  2  ADD_ITEM('ZW9W9','PVC Pipe','30-SEP-20',2,10,100.0,10000);
  3  END;
  4  /
ROW AFFECTED 1 #DATA_Affected#
OUTPUT:-

[*DUMMY RECORD*]
SQL> SELECT * FROM ITEMS;

P_COD DESCR              IN_DATE      MIN_QTY        QTY      PRICE     V_CODE
----- ------------------ --------- ---------- ---------- ---------- ----------
AB112 Power Drill        03-NOV-19          5          8     109.99      25595
ZZ999 Cordless Drill     10-JUL-20         40        200       25.5      24992
AB212 Power Drill        03-AUG-20          3         15        275      24992
Z999  PVC Pipe           30-SEP-20          2         10        100      10000

23 rows selected.
```

==========Query06==========

Write a SQL code to compile and execute the stored procedure - UPDATE_ITEM, that will update particulars (quantity and/or cost) for an item in ITEMS table with given particulars - item code, quantity of purchase, and item price. Report an error when the said item (to be updated) does not exist in ITEMS table (the NO_DATA_FOUND exception). Use the CHECK_ITEM function created earlier

==========

```
CREATE OR REPLACE PROCEDURE UPDATE_ITEM
(P_CODE1 ITEMS.P_CODE%TYPE,
QTY1 ITEMS.QTY%TYPE,
PRICE1 ITEMS.PRICE%TYPE)

IS
STATUS NUMBER(2);
BEGIN
STATUS:=CHECK_ITEM(P_CODE1);
IF STATUS=1 THEN
UPDATE ITEMS SET QTY=QTY1 WHERE
P_CODE=P_CODE1 AND QTY=QTY1 AND PRICE=PRICE1;
ELSE
RAISE NO_DATA_FOUND;
END IF;
EXCEPTION
WHEN NO_DATA_FOUND THEN
DBMS_OUTPUT.PUT_LINE('******ITEM_NOT_PRESENT IN REPOSITORY********');
END;
/

BEGIN
UPDATE_ITEM('Z999',2,202);
END;
/
[output:-]
******ITEM_NOT_PRESENT IN REPOSITORY********
```

```
====================================Query07================================
Modify procedure in Query-06, as UPDATE_ITEM_ADD_WHEN_NOT_FOUND such that when
the mentioned item is not present in ITEMS, an item is entered into ITEMS with
available particulars supplied in the procedure call. The default values for
item description, vendor code and minimum quantity as 'NEW ITEM ...', NULL and
(quantity / 8) truncated respectively. Use ADD_ITEM procedure created earlier.
You need not catch the NO_DATA_FOUND exception.
===========================================================================

SQL> CREATE SEQUENCE sequence_1
    start with 1
    increment by 1
    minvalue 0
    maxvalue 100
    ;

Sequence created.

CREATE OR REPLACE PROCEDURE UPDATE_ITEM_ADD_WHEN_NOT_FOUND
(P_CODE1 ITEMS.P_CODE%TYPE,
QTY1 ITEMS.QTY%TYPE,
PRICE1 ITEMS.PRICE%TYPE)

IS
STATUS NUMBER(2);
N PLS_INTEGER;
BEGIN
STATUS:=CHECK_ITEM(P_CODE1);
IF STATUS=1 THEN
UPDATE ITEMS SET QTY=QTY1 WHERE
P_CODE=P_CODE1 AND QTY=QTY1 AND PRICE=PRICE1;
ELSE
ADD_ITEM(P_CODE1,'NEW_ITEM-'||sequence_1.NEXTVAL,SYSDATE,2,8,PRICE1,NULL);
END IF;
END;
/

Procedure created.

SQL> BEGIN
         UPDATE_ITEM_ADD_WHEN_NOT_FOUND('aBC',2,202);
         END;
         /
ROW AFFECTED 1 #DATA_Affected#
```

```
====================================Query08================================
Write a SQL code to compile and execute the stored procedure - SHOW ITEM that
will list the item particulars for an item in ITEMS table when the item code
is supplied as input.
Report an error when the said item does not exist in ITEMS. Use the CHECK_ITEM
function created earlier.
===========================================================================
CREATE OR REPLACE PROCEDURE SHOW_ITEM(CODE ITEMS.P_CODE%TYPE)
IS
STATUS NUMBER(2);
P_CODE1 ITEMS.P_CODE%TYPE;
DESCR1 ITEMS.DESCR%TYPE;
IN_DATE1 ITEMS.IN_DATE%TYPE;
MIN_QTY1 ITEMS.MIN_QTY%TYPE;
QTY1 ITEMS.QTY%TYPE;
PRICE1 ITEMS.QTY%TYPE;
VCODE ITEMS.V_CODE%TYPE;
BEGIN
STATUS:=CHECK_ITEM(CODE);
IF STATUS=1 THEN
SELECT P_CODE,DESCR,IN_DATE,MIN_QTY,QTY,PRICE,V_CODE INTO
P_CODE1,DESCR1,IN_DATE1,MIN_QTY1,QTY1,PRICE1,VCODE
FROM ITEMS WHERE P_CODE=CODE;
DBMS_OUTPUT.PUT_LINE(P_CODE1||' '||DESCR1||' '||IN_DATE1||' '||PRICE1||'
'||VCODE);
ELSE
DBMS_OUTPUT.PUT_LINE('NO SUCH ITEM PRESENT');
END IF;
END;
/



SQL> BEGIN
        SHOW_ITEM('AB111');
       END;
        /
AB111 Power Drill 14-AUG-20 125 24992
PL/SQL procedure successfully completed.


SQL>  BEGIN
          SHOW_ITEM('aaa');
          END;
            /
NO SUCH ITEM PRESENT
```

```
==================================Query09================================

Modify the procedure in Query-08 as SHOW_ITEM_TMR_E which will handle
TOO_MANY_ROWS exception in SELECT query. In addition to exceptions in Query-06
(NO_DATA_FOUND and OTHERS) the TOO_MANY_ROWS exception should be caught when a
call to the procedure call – EXEC ADD_ITEM('HH15P', 'NEW ITEM-2',150,NULL,25);
fetches more than one row in the result set.
=========================================================================
SQL> ALTER TABLE ITEMS DROP PRIMARY KEY;
Table altered.

CREATE OR REPLACE PROCEDURE SHOW_ITEM_TMR_E (CODE ITEMS.P_CODE%TYPE)
IS
STATUS NUMBER(2);
P_CODE1 ITEMS.P_CODE%TYPE;
DESCR1 ITEMS.DESCR%TYPE;
IN_DATE1 ITEMS.IN_DATE%TYPE;
MIN_QTY1 ITEMS.MIN_QTY%TYPE;
QTY1 ITEMS.QTY%TYPE;
PRICE1 ITEMS.QTY%TYPE;
VCODE ITEMS.V_CODE%TYPE;
BEGIN
SELECT P_CODE,DESCR,IN_DATE,MIN_QTY,QTY,PRICE,V_CODE INTO
P_CODE1,DESCR1,IN_DATE1,MIN_QTY1,QTY1,PRICE1,VCODE
FROM ITEMS WHERE P_CODE=CODE;
DBMS_OUTPUT.PUT_LINE(P_CODE1||' '||DESCR1||' '||IN_DATE1||' '||MIN_QTY1||'
'||QTY1 ||' '||PRICE1||' '||VCODE);
EXCEPTION
   when too_many_rows then
    dbms_output.put_line(P_CODE1||'MULTIPLE ROWS……..');
WHEN NO_DATA_FOUND  THEN
DBMS_OUTPUT.PUT_LINE('NO SUCH ITEM PRESENT');
WHEN OTHERS THEN
DBMS_OUTPUT.PUT_LINE('SOMETHING WENT WRONG');
END;
/

SQL> exec ADD_ITEM('HH15P','NEW ITEM..','28-MAR-17',18,150,25,NULL);
ROW AFFECTED 1 #DATA_Affected#
PL/SQL procedure successfully completed

exec ADD_ITEM('HH15P','NEW ITEM-2','28-MAR-17',12,100,5.8,NULL);
ROW AFFECTED 1 #DATA_Affected#
PL/SQL procedure successfully completed.

SQL> EXEC ADD_ITEM('HH15X','Hanging Hook 15in','10-jan-13',25,200,5.75,24992);
ROW AFFECTED 1 #DATA_Affected#

SQL> BEGIN
              SHOW_ITEM_TMR_E ('HH15P');
              SHOW_ITEM_TMR_E ('HH15X');

               END;
                /
[Output]
HH15PMULTIPLE ROWS.....................
HH15X Hanging Hook 15in 10-JAN-13 25 200 5.75 24992

PL/SQL procedure successfully completed.
```

```
====================================Query10================================
 Now extend the procedure in Query-09 as SHOW_ITEM_TMR_HANDLED to print the
rows returned by the SELECT query after catching the appropriate exception.
===========================================================================
CREATE OR REPLACE PROCEDURE SHOW_ITEM_TMR_HANDLED (CODE ITEMS.P_CODE%TYPE)
IS
STATUS NUMBER(2);
P_CODE1 ITEMS.P_CODE%TYPE;
DESCR1 ITEMS.DESCR%TYPE;
IN_DATE1 ITEMS.IN_DATE%TYPE;
MIN_QTY1 ITEMS.MIN_QTY%TYPE;
QTY1 ITEMS.QTY%TYPE;
PRICE1 ITEMS.QTY%TYPE;
VCODE ITEMS.V_CODE%TYPE;
NUM NUMBER(2);
CURSOR  C1 IS SELECT P_CODE,DESCR,IN_DATE,MIN_QTY,QTY,PRICE,V_CODE INTO
P_CODE1,DESCR1,IN_DATE1,MIN_QTY1,QTY1,PRICE1,VCODE
FROM ITEMS WHERE P_CODE=CODE;
BEGIN
SELECT P_CODE,DESCR,IN_DATE,MIN_QTY,QTY,PRICE,V_CODE INTO
P_CODE1,DESCR1,IN_DATE1,MIN_QTY1,QTY1,PRICE1,VCODE
FROM ITEMS WHERE P_CODE=CODE;
DBMS_OUTPUT.PUT_LINE(P_CODE1||' '||DESCR1||' '||IN_DATE1||' '||MIN_QTY1||'
'||QTY1 ||' '||PRICE1||' '||VCODE);
EXCEPTION
   when too_many_rows then
OPEN C1;

DBMS_OUTPUT.PUT_LINE('MULTIPLE ROWS……..');
LOOP
FETCH C1 INTO P_CODE1,DESCR1,IN_DATE1,MIN_QTY1,QTY1,PRICE1,VCODE;
EXIT WHEN C1%NOTFOUND;
DBMS_OUTPUT.PUT_LINE(P_CODE1||' '||DESCR1||' '||IN_DATE1||' '||MIN_QTY1||'
'||QTY1 ||' '||PRICE1||' '||VCODE);
END LOOP;
DBMS_OUTPUT.PUT_LINE('…….……..');

CLOSE C1;
WHEN NO_DATA_FOUND  THEN
DBMS_OUTPUT.PUT_LINE('NO SUCH ITEM PRESENT');
WHEN OTHERS THEN
DBMS_OUTPUT.PUT_LINE('SOMETHING WENT WRONG');

END;
/

[OUTPUT]
SQL> BEGIN
    SHOW_ITEM_TMR_HANDLED ('HH15P');
    SHOW_ITEM_TMR_HANDLED ('HH15X');

    END;
    /
MULTIPLE ROWS.....................
HH15P NEW ITEM.. 28-MAR-17 12 100 5.8
HH15P NEW ITEM-2 28-MAR-17 18 150 25
.................................
HH15X Hanging Hook 15in 10-JAN-13 25 200 5.75 24992
***************************************************************************
```

Viva Voice
==========

**********Question-1**********
State the advantage of using storage procedure and function?
Ans:-

- Reduces network traffic This one is great advantages of PL/SQL. Because PL/SQL nature is entire block of SQL statements execute into oracle engine all at once so it's main benefit is reducing the network traffic.

- Procedural language support PL/SQL is a development tools not only for data manipulation futures but also provide the conditional checking, looping or branching operations same as like other programming language.

- Error handling PL/SQL is dealing with error handling, It's permits the smart way handling the errors and giving user friendly error messages, when the errors are encountered.

- Declare variable PL/SQL gives you control to declare variables and access them within the block. The declared variables can be used at the time of query processing.

- Intermediate Calculation Calculations in PL/SQL done quickly and efficiently without using Oracle engines. This improves the transaction performance.

- Portable application Applications are written in PL/SQL are portable in any Operating system. PL/SQL applications are independence program to run any computer.

**********Question-2**********
Explain In,Out,In Out Variable in pl/sql procedures.

Ans:-
[In]

1. Formal parameter acts like a constant: When the subprogram begins, its value is that of either its actual parameter or default value, and the subprogram cannot change this value.

2. Actual parameter can be a constant, initialized variable, literal, or expression.

3. Actual parameter is passed by reference

[Out]

1. Formal parameter is initialized to the default value of its type. The default value of the type is NULL except for a record type with a non-NULL default value.

2. When the subprogram begins, the formal parameter has its initial value regardless of the value of its actual parameter. Oracle recommends that the subprogram assign a value to the formal parameter.

3. If the default value of the formal parameter type is NULL, then the actual parameter must be a variable whose data type is not defined as NOT NULL.

4. By default, actual parameter is passed by value; if you specify NOCOPY, it might be passed by reference.

[In-Out]

1. Passes an initial value to the subprogram and returns an updated value to the invoker.

2. Formal parameter acts like an initialized variable: When the subprogram begins, its value is that of its actual parameter. Oracle recommends that the subprogram update its value.

3. Actual parameter must be a variable (typically, it is a string buffer or numeric accumulator).

4. By default, actual parameter is passed by value (in both directions); if you specify NOCOPY, it might be passed by reference.

**********Question-3**********

Differentiate between Procedure and function.

- Procedure is a way of doing things while function is the thing being done.
- Procedure is a standard way, if it changes it will be altogether another procedure the end results can be same. But if a function is changed it will be a new function completely and ther end result will be changed as well.
- A function can be performed through a Different of procedures.
- A function is objective or goal of a society or Machine while a Procedure is the way of doing things.
- A procedure may be completed without performing the function but a function can never be achieved without procedures.
- A procedure is an English literature word while functions are mathematical terms as well

**********Question-4**********

4.Write about the Raise_application_error() procedure of Oracle.

Ans:-
When the procedure raise_application_error executes, Oracle halts the execution of the current block immediately. It also reverses all changes made to the OUT or IN OUT parameters.
The changes made to the global data structure such as packaged variables, and database objects like tables will not be rolled back. Therefore, you must explicitly execute the ROLLBACK statement to reverse the effect of the DML.

****

Inference:-
==========

1. Cursor is use to retrive multiple set of data.
2. A procedure may be completed without performing the function but a function can never be achieved without procedures.
3. Formal parameter is initialized to the default value of its type. The default value of the type is NULL except for a record type with a non-NULL default value.