```
 --JOURNAL COMPILATION
/*
            AIM:To write and execute stored procedures and functions using Oracle 11g.

            PROBLEM STATEMENT:Using the relation schemata established in
                              Experiments - 01, 02, 04, and 07, create and execute the
                              mentioned stored functions and stored procedures.
*/

 ----------------------------------- QUERY-01 -----------------------------
 /*
 Write a 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)
 */
 ------------------------------------------------------------------------------
SQL> CREATE OR REPLACE PROCEDURE SHOW_EMPLOYEE(ENO IN EMPP.EID%TYPE) IS
  2  EMPREC EMPP%ROWTYPE;
  3  BEGIN
  4    SELECT * INTO EMPREC FROM EMPP WHERE EID=ENO;
  5    DBMS_OUTPUT.PUT_LINE('EID: '||EMPREC.EID||' ENAME:'||EMPREC.ENAME||' HIREDATE:'||EMPREC.HIREDATE||'
SALARY:'||EMPREC.SALARY);
  6
  7  EXCEPTION
  8  WHEN NO_DATA_FOUND THEN
  9      DBMS_OUTPUT.PUT_LINE('NO RECORD EXISTS WITH EMP ID:='||ENO);
 10  WHEN OTHERS THEN
 11      DBMS_OUTPUT.PUT_LINE(TO_CHAR(SQLCODE)||'::'||TO_CHAR(SQLERRM));
 12  END;
 13  /
Procedure created.
SQL> DECLARE
  2    ENO EMPP.EID%TYPE;
  3  BEGIN
  4    DBMS_OUTPUT.PUT_LINE('ENTER EMPLOYEE ID:');
  5    ENO:='&ENO';
  6    SHOW_EMPLOYEE(ENO);
  7  END;
  8  /

Enter value for eno: 7103
old  19:   ENO:='&ENO';
new  19:   ENO:='7103';

ENTER EMPLOYEE ID:
EID: 7103 ENAME:Julia Martin HIREDATE:01-DEC-99 SALARY:13320

PL/SQL procedure successfully completed.

 ----------------------------------- QUERY-02 -----------------------------
 /*
 Write a 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.
         7111, Justin Beiber, 01-Jan-2016, 12500
         7112, Wilfred Diesel, 02-Feb-2016, 13000
 */
 ------------------------------------------------------------------------------
SQL> CREATE OR REPLACE PROCEDURE ADD_EMPLOYEE(EMPREC EMPP%ROWTYPE) IS
  2  BEGIN
  3    INSERT INTO EMPP VALUES(EMPREC.EID,EMPREC.ENAME,EMPREC.HIREDATE,EMPREC.SALARY);
  4    DBMS_OUTPUT.PUT_LINE('RECORD INSERTED');
  5  EXCEPTION
  6    WHEN OTHERS THEN
  7        DBMS_OUTPUT.PUT_LINE(TO_CHAR(SQLCODE)||'::'||TO_CHAR(SQLERRM));
  8  END;
  9  /

Procedure created.
SQL> SELECT RPAD(OBJECT_NAME,15,' ') OBJECT_NAME,OBJECT_TYPE FROM USER_OBJECTS WHERE OBJECT_NAME='ADD_EMPLOYEE';

OBJECT_NAME     OBJECT_TYPE

--------------- -----------------------
```

```
ADD_EMPLOYEE    PROCEDURE


1 row selected.

SQL> SET SERVEROUTPUT ON;
SQL> DECLARE
  2    EMPREC EMPP%ROWTYPE;
  3  BEGIN
  4    EMPREC.EID:=7111;
  5    EMPREC.ENAME:='Justin Beiber';
  6    EMPREC.HIREDATE:='01-Jan-2016';
  7    EMPREC.SALARY:=12500;
  8    ADD_EMPLOYEE(EMPREC);
  9    EMPREC.EID:=7112;
 10    EMPREC.ENAME:='Wilfred Diesel';
 11    EMPREC.HIREDATE:='02-Feb-2016';
 12    EMPREC.SALARY:=13000;
 13    ADD_EMPLOYEE(EMPREC);
 14    COMMIT;
 15  END;
 16  /
RECORD INSERTED

RECORD INSERTED


PL/SQL procedure successfully completed.

SQL> SELECT * FROM EMPP WHERE EID IN(7111,7112);

       EID ENAME                HIREDATE      SALARY

---------- -------------------- --------- ----------

      7111 Justin Beiber        01-JAN-16      12500

      7112 Wilfred Diesel       02-FEB-16      13000


2 rows selected.

 ------------------------------------ QUERY-03 ------------------------------
 /*
 Write a 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 Upper Case) indicating employee name (use EMPP table).
 If the matching employee is not found, an appropriate exception should be raised.
 */
 ---------------------------------------------------------------------------
SQL> CREATE OR REPLACE PROCEDURE REMOVE_EMPLOYEE(EMPNAME IN EMPP.ENAME%TYPE) IS
  2    EMPID EMPP.EID%TYPE;
  3  BEGIN
  4    SELECT EID INTO EMPID FROM EMPP WHERE UPPER(ENAME)=UPPER(EMPNAME);
  5    DELETE FROM EMPP WHERE EID=EMPID;
  6    DBMS_OUTPUT.PUT_LINE('RECORD DELETED');
  7  EXCEPTION
  8  WHEN NO_DATA_FOUND THEN
  9      DBMS_OUTPUT.PUT_LINE('NO RECORD FOR '||EMPNAME);
 10  WHEN OTHERS THEN
 11      DBMS_OUTPUT.PUT_LINE(TO_CHAR(SQLCODE)||'::'||TO_CHAR(SQLERRM));
 12  END;
 13  /

Procedure created.

SQL> EXEC REMOVE_EMPLOYEE('PRAKHAR SRIVASTAVA');
NO RECORD FOR PRAKHAR SRIVASTAVA


PL/SQL procedure successfully completed.

SQL> EXEC REMOVE_EMPLOYEE('Wilfred Diesel');
RECORD DELETED
```

```
PL/SQL procedure successfully completed.

SQL> ROLLBACK;

Rollback complete.

 ---------------------------------- QUERY-04 ------------------------------
 /*
 Write a 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> DROP TABLE ITEMS;

Table dropped.

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

Table created.

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

Table altered.

SQL> ALTER TABLE ITEMS MODIFY IN_DATE  DEFAULT SYSDATE;

Table altered.

SQL> ALTER TABLE ITEMS MODIFY MIN_QTY  DEFAULT 2;

Table altered.

SQL> CREATE OR REPLACE FUNCTION CHECK_ITEM(PCODE ITEMS.P_CODE%TYPE)
  2  RETURN NUMBER IS
  3  CNT NUMBER;
  4  N INT(1) :=1 ;
  5  BEGIN
  6    SELECT COUNT(*) INTO CNT FROM ITEMS WHERE P_CODE=PCODE;
  7    IF CNT>=1 THEN
  8      RETURN N;
  9    ELSE
 10      N:=0;
 11      RETURN N;
 12    END IF;
 13  END;
 14  /

Function created.

SQL> BEGIN
  2    DBMS_OUTPUT.PUT_LINE('STATUS: '||CHECK_ITEM('JB008'));
  3    DBMS_OUTPUT.PUT_LINE('STATUS: '||CHECK_ITEM('JB555'));
  4  END;
  5  /
STATUS: 1

STATUS: 0


PL/SQL procedure successfully completed.

 ---------------------------------- QUERY-05 ------------------------------
 /*
 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.
 */
 ------------------------------------------------------------------------------
SQL> CREATE OR REPLACE FUNCTION IREC_CREATOR(ICODE ITEMS.P_CODE%TYPE,IDESC ITEMS.DESCR%TYPE,IQTY
ITEMS.QTY%TYPE,IPRICE ITEMS.PRICE%TYPE,VCODE ITEMS.V_CODE%TYPE)
  2  RETURN ITEMS%ROWTYPE IS
```

```
  3  IREC ITEMS%ROWTYPE;
  4  BEGIN
  5      IREC.P_CODE:=ICODE;
  6      IREC.DESCR:=IDESC;
  7      IREC.IN_DATE:=SYSDATE;
  8      IREC.MIN_QTY:=NULL;
  9      IREC.QTY:=IQTY;
 10      IREC.PRICE:=IPRICE;
 11      IREC.V_CODE:=NULL;
 12    RETURN IREC;
 13  END;
 14  /

Function created.

SQL> CREATE OR REPLACE PROCEDURE ADD_ITEM(IREC IN ITEMS%ROWTYPE)IS
  2  BEGIN
  3    INSERT INTO ITEMS(P_CODE,DESCR,QTY,PRICE,V_CODE)
VALUES(IREC.P_CODE,IREC.DESCR,IREC.QTY,IREC.PRICE,IREC.V_CODE);
  4    DBMS_OUTPUT.PUT_LINE('RECORD INSERTED');
  5  END;
  6  /

Procedure created.

SQL> EXEC ADD_ITEM(IREC_CREATOR('YJ304','DIGGER',6,999.99,25595));
RECORD INSERTED


PL/SQL procedure successfully completed.

 ----------------------------------- QUERY-06 ------------------------------
 /*
 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.
 */
 ------------------------------------------------------------------------------
SQL> CREATE OR REPLACE PROCEDURE UPDATE_ITEM (ICODE IN ITEMS.P_CODE%TYPE,IQTY IN ITEMS.QTY%TYPE,IPRICE IN
ITEMS.PRICE%TYPE) IS
  2  BEGIN
  3    IF CHECK_ITEM(ICODE)=1 THEN
  4      UPDATE ITEMS SET QTY= IQTY, PRICE=IPRICE WHERE P_CODE=ICODE;
  5      DBMS_OUTPUT.PUT_LINE('RECORD UPDATED');
  6    ELSE
  7      RAISE NO_DATA_FOUND;
  8    END IF;
  9  EXCEPTION
 10    WHEN NO_DATA_FOUND THEN
 11      DBMS_OUTPUT.PUT_LINE('NO DATA FOUND FOR ITEM CODE: '||ICODE);
 12    WHEN OTHERS THEN
 13      DBMS_OUTPUT.PUT_LINE(TO_CHAR(SQLCODE)||'::'||TO_CHAR(SQLERRM));
 14  END;
 15  /

Procedure created.

SQL> EXEC UPDATE_ITEM('YJ404',10,599.99);
NO DATA FOUND FOR ITEM CODE: YJ404


PL/SQL procedure successfully completed.

SQL> EXEC UPDATE_ITEM('YJ304',10,599.99);
RECORD UPDATED


PL/SQL procedure successfully completed.

 ----------------------------------- QUERY-07 ------------------------------
 /*
 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.
 */
 --------------------------------------------------------------------------------
SQL> CREATE OR REPLACE PROCEDURE UPDATE_ITEM_ADD_WHEN_NOT_FOUND (ICODE IN ITEMS.P_CODE%TYPE,IQTY IN
ITEMS.QTY%TYPE,IPRICE IN ITEMS.PRICE%TYPE) IS
  2  IREC ITEMS%ROWTYPE;
  3  BEGIN
  4    IF CHECK_ITEM(ICODE)=1 THEN
  5      UPDATE ITEMS SET QTY= IQTY, PRICE=IPRICE WHERE P_CODE=ICODE;
  6      DBMS_OUTPUT.PUT_LINE('RECORD UPDATED');
  7    ELSE
  8        IREC.P_CODE:=ICODE;
  9        IREC.DESCR:='NEW ITEM';
 10        IREC.IN_DATE:=SYSDATE;
 11        IREC.MIN_QTY:=IQTY/8;
 12        IREC.QTY:=IQTY;
 13        IREC.PRICE:=IPRICE;
 14        IREC.V_CODE:=NULL;
 15        ADD_ITEM(IREC);
 16    END IF;
 17  END;
 18  /

Procedure created.

SQL> EXEC UPDATE_ITEM_ADD_WHEN_NOT_FOUND('YJ404',10,599.99);
RECORD INSERTED


PL/SQL procedure successfully completed.

 ---------------------------------- QUERY-08 ------------------------------
 /*
 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 to be updated does not exist in ITEMS. Use the CHECK_ITEM
 function created earlier.
 */
 --------------------------------------------------------------------------------
SQL> CREATE OR REPLACE PROCEDURE SHOW_ITEM(ICODE ITEMS.P_CODE%TYPE) IS
  2  IREC ITEMS%ROWTYPE;
  3  BEGIN
  4    IF CHECK_ITEM(ICODE)=1 THEN
  5        SELECT * INTO IREC FROM ITEMS WHERE P_CODE=ICODE;
  6        DBMS_OUTPUT.PUT_LINE(IREC.P_CODE||' '||RPAD(IREC.DESCR,15,' ')||' '||IREC.IN_DATE||'
'||IREC.MIN_QTY||' '||IREC.QTY||' '||IREC.PRICE||' '||IREC.V_CODE);
  7    ELSE
  8        RAISE NO_DATA_FOUND;
  9    END IF;
 10  EXCEPTION
 11    WHEN NO_DATA_FOUND THEN
 12      DBMS_OUTPUT.PUT_LINE(ICODE||' NO DATA FOUND');
 13  END;
 14  /

Procedure created.

SQL> BEGIN
  2  SHOW_ITEM('HH15P');
  3  SHOW_ITEM('HH15X');
  4  SHOW_ITEM('YJ304');
  5  SHOW_ITEM('SHU00');
  6  SHOW_ITEM('SH100');
  7  SHOW_ITEM('MP100');
  8  END;
  9  /
HH15P NO DATA FOUND

HH15X HANGING HOOK 15 10-JAN-13 25 200 5.75 24992

YJ304 DIGGER          14-APR-17 2 10 599.99

SHU00 NO DATA FOUND

SH100 Sledge Hammer   02-JAN-12 5 8 14.4
```

```
MP100 NO DATA FOUND


PL/SQL procedure successfully completed.

  ---------------------------------- QUERY-09 ------------------------------
 /*
 Modify the procedure in Query-08 as SHOW_ITEM_TMR_E which will handle TOO_MANY_ROWS
 exception in SELECT query. In addition to the exceptions in Query-06 (NO_DATA_FOUND and OTHERS)
 the TOO_MANY_ROWS exception should be caught as the procedure call –
 EXEC ADD_ITEM('HH15P','NEW ITEM-2',150,NULL,25); will fetch more than one row in the result set.
 */
 ------------------------------------------------------------------------------
SQL> ALTER TABLE ITEMS DROP PRIMARY KEY;

Table altered.

SQL> EXEC ADD_ITEM(IREC_CREATOR('HH15P','NEW ITEM ..',150,00,25));
RECORD INSERTED


PL/SQL procedure successfully completed.

SQL> EXEC ADD_ITEM(IREC_CREATOR('HH15P','NEW ITEM-2',150,00,25));
RECORD INSERTED


PL/SQL procedure successfully completed.

SQL> CREATE OR REPLACE PROCEDURE SHOW_ITEM_TMR_HANDLED(ICODE ITEMS.P_CODE%TYPE) IS
  2  IREC ITEMS%ROWTYPE;
  3  BEGIN
  4    IF CHECK_ITEM(ICODE)=1 THEN
  5        SELECT * INTO IREC FROM ITEMS WHERE P_CODE=ICODE;
  6        DBMS_OUTPUT.PUT_LINE(IREC.P_CODE||' '||RPAD(IREC.DESCR,15,' ')||' '||IREC.IN_DATE||'
'||IREC.MIN_QTY||' '||IREC.QTY||' '||IREC.PRICE||' '||IREC.V_CODE);
  7    ELSE
  8        RAISE NO_DATA_FOUND;
  9    END IF;
 10  EXCEPTION
 11    WHEN TOO_MANY_ROWS THEN
 12      DBMS_OUTPUT.PUT_LINE('MULTIPLE ROWS DETECTED FOR ITEM CODE : '||ICODE);
 13    WHEN NO_DATA_FOUND THEN
 14      DBMS_OUTPUT.PUT_LINE(ICODE||' NO DATA FOUND');
 15    WHEN OTHERS THEN
 16      DBMS_OUTPUT.PUT_LINE(TO_CHAR(SQLCODE)||'::'||TO_CHAR(SQLERRM));
 17  END;
 18  /

Procedure created.

SQL> BEGIN
  2  SHOW_ITEM_TMR_HANDLED('HH15P');
  3  SHOW_ITEM_TMR_HANDLED('HH15X');
  4  END;
  5  /
MULTIPLE ROWS DETECTED FOR ITEM CODE : HH15P

HH15X NO DATA FOUND


PL/SQL procedure successfully completed.

 ---------------------------------- QUERY-10 ------------------------------
 /*
 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.
 */
 ------------------------------------------------------------------------------
SQL> CREATE OR REPLACE PROCEDURE SHOW_ITEM_TMR_HANDLED(ICODE ITEMS.P_CODE%TYPE) IS
  2  IREC ITEMS%ROWTYPE;
  3  CURSOR IRECS IS SELECT * FROM ITEMS WHERE P_CODE=ICODE;
  4  BEGIN
  5    IF CHECK_ITEM(ICODE)=1 THEN
```

```
  6        SELECT * INTO IREC FROM ITEMS WHERE P_CODE=ICODE;
  7        DBMS_OUTPUT.PUT_LINE(IREC.P_CODE||' '||RPAD(IREC.DESCR,15,' ')||' '||IREC.IN_DATE||'
'||IREC.MIN_QTY||' '||IREC.QTY||' '||IREC.PRICE||' '||IREC.V_CODE);
  8    ELSE
  9        RAISE NO_DATA_FOUND;
 10    END IF;
 11  EXCEPTION
 12    WHEN TOO_MANY_ROWS THEN
 13      DBMS_OUTPUT.PUT_LINE('ERROR MULTIPLE ROWS DETECTED FOR ITEM CODE : '||ICODE);
 14      OPEN IRECS;
 15      LOOP
 16      FETCH IRECS INTO IREC;
 17      EXIT WHEN IRECS % NOTFOUND;
 18      DBMS_OUTPUT.PUT_LINE(IREC.P_CODE||' '||RPAD(IREC.DESCR,15,' ')||' '||IREC.IN_DATE||' '||IREC.MIN_QTY||'
'||IREC.QTY||' '||IREC.PRICE||' '||IREC.V_CODE);
 19      END LOOP;
 20      CLOSE IRECS;
 21    WHEN NO_DATA_FOUND THEN
 22      DBMS_OUTPUT.PUT_LINE(ICODE||' NO DATA FOUND');
 23    WHEN OTHERS THEN
 24      DBMS_OUTPUT.PUT_LINE(TO_CHAR(SQLCODE)||'::'||TO_CHAR(SQLERRM));
 25  END;
 26  /

Procedure created.

SQL> BEGIN
  2  SHOW_ITEM_TMR_HANDLED('HH15P');
  3  SHOW_ITEM_TMR_HANDLED('HH15X');
  4  END;
  5  /
ERROR MULTIPLE ROWS DETECTED FOR ITEM CODE : HH15P

HH15P NEW ITEM ..     14-APR-17 2 150 0

HH15P NEW ITEM-2      14-APR-17 2 150 0

HH15X HANGING HOOK 15 10-JAN-13 25 200 5.75 24992


PL/SQL procedure successfully completed.

 ---------------------------------- END OF QUERIES---------------------------

SQL> SET FEEDBACK OFF
SQL> SPOOL OFF
```