-------  
set serveroutput on;  
-- creation of tic tac toe table  
DECLARE  
NUM1 NUMBER;  
BEGIN  
SELECT count(\*) INTO NUM1 FROM USER\_TABLES  
WHERE TABLE\_NAME = 'T\_T\_TABLE';  
IF NUM1 != 0 THEN  
EXECUTE IMMEDIATE 'DROP TABLE T\_T\_Table purge';  
END IF;  
EXECUTE IMMEDIATE 'CREATE TABLE T\_T\_Table(  
N NUMBER,  
I VARCHAR2(1000),  
J VARCHAR2(1000),  
K VARCHAR2(1000))';  
END;  
/  
---------------------------------------------------------------------  
-- Previous play  
CREATE OR REPLACE PACKAGE tic\_tac\_package IS  
previous\_play char(1) := '\_';  
is\_reset boolean := FALSE;  
END;  
/  
-- NUMBER TO COLUMN NAME CONVERSION  
CREATE OR REPLACE FUNCTION NUM\_NAME(n1 IN NUMBER)  
RETURN varCHAR2  
IS  
BEGIN  
IF n1=1 THEN  
RETURN 'I';  
ELSIF n1=2 THEN  
RETURN 'J';  
ELSIF n1=3 THEN  
RETURN 'K';  
ELSE  
RETURN '\_';  
END IF;  
END;  
/  
-----------------------------------------------------------------------------------  
--PROCEDURE TO DISPLAY GAME BOARD  
CREATE OR REPLACE PROCEDURE DISPLAY IS  
BEGIN  
DBMS\_OUTPUT.ENABLE(10000);  
dbms\_output.put\_line(' ');  
FOR ll in (SELECT \* FROM T\_T\_Table ORDER BY N) LOOP  
dbms\_output.put\_line(' ' || ll.I || ' ' || ll.J || ' ' || ll.K);  
END LOOP;  
dbms\_output.put\_line(' ');  
END;  
/  
-----------------------------------------------------------------------------------  
--GAME RESET PROCEDURE  
CREATE OR REPLACE PROCEDURE RESET1 IS  
ii NUMBER;  
BEGIN  
EXECUTE IMMEDIATE 'TRUNCATE TABLE T\_T\_Table';  
FOR ii in 1..3 LOOP  
INSERT INTO T\_T\_Table VALUES (ii,'\_','\_','\_');  
END LOOP;  
tic\_tac\_package.previous\_play := '\_';  
dbms\_output.enable(10000);  
DISPLAY();  
dbms\_output.put\_line('ARE YOU READY TO PLAY: EXECUTE PLAY("X", col, row);');  
tic\_tac\_package.is\_reset := FALSE;  
END;  
/  
-------------------------------------------------------------------------------------  
-- PROCEDURE TO PLAY  
CREATE OR REPLACE PROCEDURE INVALID\_PLAY(SYM IN VARCHAR2, COL IN NUMBER) IS  
NOT\_A\_VALID\_MOVE EXCEPTION;  
NOT\_A\_VALID\_COLUMN EXCEPTION;  
NOT\_A\_VALID\_ROW EXCEPTION;  
PRAGMA EXCEPTION\_INIT(NOT\_A\_VALID\_MOVE,-20000);  
BEGIN  
IF SYM NOT IN ('X','O') THEN  
RAISE\_APPLICATION\_ERROR(-20000,'NOT A VALID MOVE',FALSE);  
END IF;  
END;  
/  
-----------------------  
--INVALID COLUMN  
CREATE OR REPLACE PROCEDURE INVALID\_COL(V IN NUMBER) IS  
NOT\_A\_VALID\_COLUMN EXCEPTION;  
PRAGMA EXCEPTION\_INIT(NOT\_A\_VALID\_COLUMN, -20000);  
BEGIN  
IF V NOT IN (1,2,3) THEN  
RAISE\_APPLICATION\_ERROR(-20000,'NOT A VALID COLUMN');  
END IF;  
END;  
/  
-----PROCEDURE FOR MAIN GAME-----  
CREATE OR REPLACE PROCEDURE PLAY(sym IN VARCHAR2, colnb IN NUMBER, lig IN NUMBER) IS  
val T\_T\_Table.i%TYPE;  
colo CHAR(2);  
sym2 CHAR(2);  
BEGIN  
SELECT NUM\_NAME(colnb) INTO colo FROM DUAL;  
BEGIN  
if tic\_tac\_package.is\_reset = TRUE then  
RESET1();  
end if;

if tic\_tac\_package.previous\_play = '\_' then  
tic\_tac\_package.previous\_play := sym;  
elsif tic\_tac\_package.previous\_play = 'X' and sym = 'X' then  
RAISE\_APPLICATION\_ERROR(-20000,'Foul play...!');  
elsif tic\_tac\_package.previous\_play = 'O' and sym = 'O' then  
RAISE\_APPLICATION\_ERROR(-20000,'Foul play...!');  
end if;

INVALID\_COL(COLNB);

EXECUTE IMMEDIATE ('SELECT ' || colo || ' FROM T\_T\_Table WHERE N=' || lig) INTO val;  
EXCEPTION  
WHEN NO\_DATA\_FOUND THEN  
RAISE\_APPLICATION\_ERROR(-20000,'OUT OF RANGE VALUE');  
END;  
IF val='\_' THEN  
EXECUTE IMMEDIATE ('UPDATE T\_T\_Table SET ' || colo || '=''' || sym || ''' WHERE N=' || lig);  
INVALID\_PLAY(SYM,LIG);  
ELSE  
dbms\_output.enable(10000);  
dbms\_output.put\_line('You cannot play this square, it is already played');  
END IF;  
tic\_tac\_package.previous\_play := sym;  
END;  
/  
--------------------------------------------------------------------------------  
-- PROCEDURE TO WIN - CHAMPION  
CREATE OR REPLACE PROCEDURE WIN1(SYM IN VARCHAR2) IS  
BEGIN  
dbms\_output.enable(10000);  
tic\_tac\_package.is\_reset := TRUE;  
if SYM = '\_' then  
dbms\_output.put\_line('It is a draw! No one won!!');  
else  
dbms\_output.put\_line('Player ' || SYM || ' won!!');  
end if;  
dbms\_output.put\_line('--------------------------------------');  
END;  
/  
----------------------------------------------------------------------------------  
--col  
CREATE OR REPLACE FUNCTION COL1(nomcol IN VARCHAR2, SYM IN VARCHAR2)  
RETURN varCHAR2  
IS  
P number;  
BEGIN  
EXECUTE IMMEDIATE ('SELECT count(\*) FROM T\_T\_Table WHERE '|| nomcol || '=''' || SYM || '''') INTO P;  
RETURN P;  
END;  
/  
--------------------------------------------------------------------------------------  
-- COLUMN TEST FUNCTION - columnwin  
CREATE OR REPLACE FUNCTION COL3(nomcol IN VARCHAR2)  
RETURN VARCHAR2  
IS  
tmpx1 NUMBER;  
r VARCHAR2(1);  
BEGIN  
-- SELECT COL1(nomcol, 'X') into r FROM DUAL;  
tmpx1:=COL1(nomcol, 'X');  
-- EXECUTE IMMEDIATE r INTO tmpx1;  
IF tmpx1=3 THEN  
RETURN 'X';  
end if;  
-- SELECT COL1(nomcol, 'O') into r FROM DUAL;  
tmpx1:=COL1(nomcol, 'O');  
-- EXECUTE IMMEDIATE r INTO tmpx1;  
IF tmpx1=3 THEN  
RETURN 'O';  
END IF;

RETURN '\_';  
END;  
/  
--select COL3('x') FROM DUAL;-----------------------------------------------------------------------------------------------------------------------------  
-- diagonal test function ---- diagonal win  
CREATE OR REPLACE FUNCTION COL4(t\_char IN CHAR)  
RETURN char  
IS  
char\_count\_l number := 0;  
char\_count\_r number := 0;  
BEGIN  
EXECUTE IMMEDIATE ('SELECT count(\*) FROM T\_T\_Table WHERE (N=1 and I=''' || t\_char || ''') or (N=2 and J=''' || t\_char || ''') or (N=3 and K=''' || t\_char || ''')') INTO char\_count\_l;  
EXECUTE IMMEDIATE ('SELECT count(\*) FROM T\_T\_Table WHERE (N=3 and I=''' || t\_char || ''') or (N=2 and J=''' || t\_char || ''') or (N=1 and K=''' || t\_char || ''')') INTO char\_count\_r;  
if char\_count\_r = 3 or char\_count\_l = 3 then  
RETURN t\_char;  
end if;  
return '\_';  
END;  
/  
------------------------------------------------------------  
-- test trigger if we win  
CREATE OR REPLACE TRIGGER winn  
AFTER UPDATE ON T\_T\_Table  
DECLARE  
CURSOR cr\_lig IS  
SELECT \* FROM T\_T\_Table ORDER BY N;  
crlv T\_T\_Table%rowtype;  
blank\_count number;  
tmpvar1 CHAR;  
tmpx1 CHAR;  
tmpx2 CHAR;  
r VARCHAR2(40);  
win\_num number := 0;  
BEGIN  
FOR crlv IN cr\_lig LOOP  
-- line test  
IF crlv.I = crlv.J AND crlv.J = crlv.K AND NOT crlv.I='\_' THEN  
WIN1(crlv.I);  
win\_num := 1;  
EXIT;  
END IF;  
-- colon test  
SELECT COL3(NUM\_NAME(crlv.N)) INTO tmpvar1 FROM DUAL;  
IF NOT tmpvar1 = '\_' THEN  
WIN1(tmpvar1);  
win\_num := 1;  
EXIT;  
END IF;  
END LOOP;  
-- diagonal test  
tmpx1 := COL4('X');  
tmpx2 := COL4('O');  
IF NOT tmpx1 = '\_' THEN  
WIN1(tmpx1);  
win\_num := 1;  
END IF;  
IF NOT tmpx2 = '\_' THEN  
WIN1(tmpx2);  
win\_num := 1;  
END IF;

-- draw scenario  
SELECT COUNT(\*) into blank\_count FROM T\_T\_Table WHERE I='\_' or J='\_' or K='\_';  
if blank\_count = 0 then  
WIN1('\_');  
win\_num := 1;  
end if;  
DISPLAY();  
if win\_num != 0 then  
dbms\_output.put\_line('Game is over... Please reset data...!');  
end if;  
--dbms\_output.put\_line('RESET' || sys.diutil.bool\_to\_int(tic\_tac\_package.is\_reset));  
END;  
/