**DMDD Assignment 3: Tic – Tac – Toe Game**

By: Aditya Agarwal

Github Link - https://github.com/adiag321/Tic-Tac-Toe-Game-Using-PL-SQL

**Rules:**

1. The Row number and column number are indexed with “1”, “2” and “3” and is converted into “A”, “B” and “C”.
2. In this game, all the values are manually inserted, and 4 test cases are implemented
3. “X” and “O” values are inserted in Alternate Fashion.
4. If user inserts values other than “X” and “O”, then an error is shown in the command terminal.
5. If the user inserts row value and column value other than “1”, “2” and “3”, then an error is shown accordingly to the user in the command line.
6. The user needs to restart the game by “Resetting” it and calling the “Resetting\_game” procedure.

**Code:**

--drop table tic\_tac\_toe\_game;

set serveroutput on;

CREATE TABLE tic\_tac\_toe\_game(

Y NUMBER,

A CHAR,

B CHAR,

C CHAR

);

select \* from tic\_tac\_toe\_game;

CREATE OR REPLACE FUNCTION num\_to\_col (num IN NUMBER)

RETURN CHAR IS

BEGIN

IF num = 1 THEN

RETURN 'A';

ELSIF num = 2 THEN

RETURN 'B';

ELSIF num = 3 THEN

RETURN 'C';

ELSE

RETURN '\_';

END IF;

END;

CREATE OR REPLACE PROCEDURE show\_matrix IS

BEGIN

dbms\_output.put\_line(' ');

FOR i in (SELECT \* FROM tic\_tac\_toe\_game ORDER BY Y) LOOP

dbms\_output.put\_line(' ' || i.A || ' ' || i.B || ' ' || i.C);

END LOOP;

dbms\_output.put\_line(' ');

END;

CREATE OR REPLACE PROCEDURE reseting\_game IS

i NUMBER;

BEGIN

DELETE FROM tic\_tac\_toe\_game;

FOR i in 1..3 LOOP

INSERT INTO tic\_tac\_toe\_game VALUES (i,'\_','\_','\_');

END LOOP;

show\_matrix();

dbms\_output.put\_line('New Game : EXECUTE game(''X'', x, y);');

END;

CREATE OR REPLACE PROCEDURE show\_winner (Character\_value IN VARCHAR2) IS

BEGIN

show\_matrix();

dbms\_output.put\_line('Player ' || Character\_value || ' Won the Game.');

dbms\_output.put\_line('To start a new game, RESET the game');

END;

CREATE OR REPLACE PROCEDURE playing\_game(Character\_value IN VARCHAR2, row\_value IN NUMBER, col\_value IN NUMBER) IS

val tic\_tac\_toe\_game.a%type;

cols CHAR;

Character\_value2 CHAR;

A1 tic\_tac\_toe\_game.a%type;

A2 tic\_tac\_toe\_game.a%type;

A3 tic\_tac\_toe\_game.a%type;

B1 tic\_tac\_toe\_game.a%type;

B2 tic\_tac\_toe\_game.a%type;

B3 tic\_tac\_toe\_game.a%type;

C1 tic\_tac\_toe\_game.a%type;

C2 tic\_tac\_toe\_game.a%type;

C3 tic\_tac\_toe\_game.a%type;

-- Exception

condition\_tie exception;

wrong\_row\_no exception;

wrong\_column\_no exception;

wrong\_symbol exception;

FLAG BOOLEAN:=TRUE;

BEGIN

-- Handling irregular column value, row value and character

IF col\_value < 1 OR col\_value > 3 then

raise wrong\_column\_no;

END IF;

IF row\_value < 1 OR row\_value > 3 then

raise wrong\_row\_no;

END IF;

IF Character\_value <> 'O' AND Character\_value <> 'X' then

raise wrong\_symbol;

END IF;

SELECT num\_to\_col (col\_value) INTO cols FROM DUAL;

EXECUTE IMMEDIATE ('SELECT ' || cols || ' FROM tic\_tac\_toe\_game WHERE y=' || row\_value) INTO val;

IF val='\_' THEN

EXECUTE IMMEDIATE ('UPDATE tic\_tac\_toe\_game SET ' || cols || '=''' || Character\_value || ''' WHERE y=' || row\_value);

IF Character\_value = 'X' THEN

Character\_value2 := 'O';

ELSE

Character\_value2 := 'X';

END IF;

show\_matrix();

EXECUTE IMMEDIATE ('SELECT A FROM tic\_tac\_toe\_game WHERE y=1') INTO A1;

EXECUTE IMMEDIATE ('SELECT B FROM tic\_tac\_toe\_game WHERE y=1') INTO B1;

EXECUTE IMMEDIATE ('SELECT C FROM tic\_tac\_toe\_game WHERE y=1') INTO C1;

EXECUTE IMMEDIATE ('SELECT A FROM tic\_tac\_toe\_game WHERE y=2') INTO A2;

EXECUTE IMMEDIATE ('SELECT B FROM tic\_tac\_toe\_game WHERE y=2') INTO B2;

EXECUTE IMMEDIATE ('SELECT C FROM tic\_tac\_toe\_game WHERE y=2') INTO C2;

EXECUTE IMMEDIATE ('SELECT A FROM tic\_tac\_toe\_game WHERE y=3') INTO A3;

EXECUTE IMMEDIATE ('SELECT B FROM tic\_tac\_toe\_game WHERE y=3') INTO B3;

EXECUTE IMMEDIATE ('SELECT C FROM tic\_tac\_toe\_game WHERE y=3') INTO C3;

IF (A1=C1) AND (A1=B1) AND A1 <> '\_' THEN

show\_winner(A1);

FLAG:=FALSE;

END IF;

IF (A2=C2) AND (A2=B2) AND A2 <> '\_' THEN

show\_winner(A2);

FLAG:=FALSE;

END IF;

IF (A3=C3) AND (A3=B3) AND A3 <> '\_' THEN

show\_winner(A3);

FLAG:=FALSE;

END IF;

IF (A1=A3) AND (A1=A2) AND A1 <> '\_' THEN

show\_winner(A1);

FLAG:=FALSE;

END IF;

IF (B1=B3) AND (B1=B2) AND B1 <> '\_' THEN

show\_winner(B1);

FLAG:=FALSE;

END IF;

IF (C1=C3) AND (C1=C2) AND C1 <> '\_' THEN

show\_winner(C1);

FLAG:=FALSE;

END IF;

IF (A1=C3) AND (A1=B2) AND A1 <> '\_' THEN

show\_winner(A1);

FLAG:=FALSE;

END IF;

IF (C1=A3) AND (C1=B2) AND C1 <> '\_' THEN

show\_winner(C1);

FLAG:=FALSE;

END IF;

IF B1 <> '\_' AND B2 <> '\_' AND B3 <> '\_' AND C1 <> '\_' AND C2 <> '\_' AND C3 <> '\_' AND A1 <> '\_' AND A2 <> '\_' AND A3 <> '\_' THEN

raise condition\_tie;

END IF;

IF FLAG THEN

dbms\_output.put\_line('Next turn ' || Character\_value2 || ' to play : EXECUTE game(''' || Character\_value2 || ''', x, y);');

END IF;

ELSE

dbms\_output.put\_line('Other player has played on this slot. Try on different slot');

END IF;

-- Exeption Handling

EXCEPTION

WHEN wrong\_symbol THEN

dbms\_output.put\_line('Given Symbol is not "X" or "O"');

--reseting\_game();

WHEN condition\_tie THEN

dbms\_output.put\_line('It is a Tie');

--reseting\_game();

WHEN wrong\_column\_no THEN

dbms\_output.put\_line('Given Column Number should be in the range of 1 to 3');

reseting\_game();

WHEN wrong\_row\_no THEN

dbms\_output.put\_line('Given Row Number should be in the range of 1 to 3');

--reseting\_game();

END;

**TEST CASES:**

1. Test Case 1 :

EXECUTE reseting\_game;

EXECUTE playing\_game('O', 3, 1);

EXECUTE playing\_game('X', 2, 2);

EXECUTE playing\_game('O', 1, 1);

EXECUTE playing\_game('X', 2, 3);

EXECUTE playing\_game('O', 2, 1);

Text

Description automatically generated with low confidence

1. Test Case 2: Tie Condition

EXECUTE reseting\_game;

EXECUTE playing\_game('O', 1, 3);

EXECUTE playing\_game('X', 1, 1);

EXECUTE playing\_game('O', 2, 2);

EXECUTE playing\_game('X', 3, 1);

EXECUTE playing\_game('O', 2, 1);

EXECUTE playing\_game('X', 3, 2);

EXECUTE playing\_game('O', 2, 3);

EXECUTE playing\_game('X', 1, 2);

EXECUTE playing\_game('O', 3, 3);

Text

Description automatically generated

1. Test Case 3: UNKNOWN ROW NUMBER AND COLUMN NUMBER

EXECUTE reseting\_game;

EXECUTE playing\_game('X', 1, 3);

EXECUTE playing\_game('O', 2, 4);

EXECUTE playing\_game('X', 3, 1);

EXECUTE playing\_game('O', 0, 1);

Text

Description automatically generated

1. Test Case 4: WHEN PLAYER INPUTS CHARACTER OTHER THAN "X" AND "O"

EXECUTE reseting\_game;

EXECUTE playing\_game('X', 1, 3);

EXECUTE playing\_game('A', 2, 1);

EXECUTE playing\_game('X', 1, 2);

EXECUTE playing\_game('O', 2, 3);

EXECUTE playing\_game('P', 1, 1);

Text

Description automatically generated with low confidence