DECLARE

numoftables NUMBER;

BEGIN

SELECT count(\*) INTO numoftables FROM user\_tables

WHERE TABLE\_NAME = 'TicTacToeGame';

IF numoftables = 0 THEN

EXECUTE IMMEDIATE 'CREATE TABLE TicTacToeGame(

row\_num NUMBER,

C1 char,

C2 char,

C3 char

)';

END IF;

END;

/\*DECLARE

t NUMBER;

BEGIN

SELECT count(\*) INTO t FROM user\_tables

WHERE TABLE\_NAME = 'pt';

IF t = 0 THEN

EXECUTE IMMEDIATE 'CREATE TABLE pt(

turn varchar2(1)

)';

END IF;

END;

drop table if exists playert;

create table playert (turn varchar2(1) not null);

insert into playert (turn) values ('X');

drop table if exists playerturns;

create table playerturns (turn varchar2);

insert into playerturns(turn) values ('X');

DROP TABLE IF EXISTS PlayerTurns;

CREATE TABLE PlayerTurns (turn VARCHAR2 NOT NULL);

INSERT INTO PlayerTurns (turn) VALUES ('X');\*/

DROP TABLE IF EXISTS ttt\_PlayerTurn2;

CREATE TABLE ttt\_PlayerTurn2 (turn CHAR);

INSERT INTO ttt\_PlayerTurn2 (turn) VALUES ('X');

set serveroutput on;

create or replace function colname(num in number)

return char

is

begin

if num=1 then return 'C1';

elsif num=2 then return 'C2';

elsif num=3 then return 'C3';

else return '\_';

end if;

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

end;

/

set serveroutput on;

create or replace procedure print\_gameboard is

begin

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

dbms\_output.enable(10000);

for linnum in (select \* from TicTacToeGame order by row\_num) loop

dbms\_output.put\_line(' '||linnum.C1||' '||linnum.C2||' '||linnum.C3);

end loop;

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

end;

/

set serveroutput on;

create or replace procedure reset\_gameboard is v number;

begin

delete from TicTacToeGame;

for v in 1..3 loop

insert into TicTacToeGame values(v,'\_','\_','\_');

end loop;

update ttt\_PlayerTurn2 SET turn='X';

dbms\_output.enable(10000);

print\_gameboard();

dbms\_output.put\_line('The game is ready to play:execute play("X",x,y);');

end;

/

create or replace procedure playgame(sym in char,col in number,rownum in number) is

v TicTacToe.a%type;

colu char;

sym2 char;

playerturn ttt\_PlayerTurn2.turn%type;

begin

select colname(col) into colu from dual;

select turn into playerturn from ttt\_PlayerTurn2;

execute immediate('select'||colu||'from TicTacToe where row\_num='||rownum) into v;

if sym!= playerturn then dbms\_output.put\_line('You cant play this turn');

else

if v='\_' then execute immediate('update TicTacToe set'||colu||'='''||sym||'''where row\_num='||rownum);

if sym='X' then sym2:='O';

update ttt\_PlayerTurn2 set ttt\_PlayerTurn2.turn=sym2;

else

sym2:='X';

update ttt\_PlayerTurn2 set ttt\_PlayerTurn2.turn=sym2;

end if;

print\_gameboard();

dbms\_output.put\_line(sym2||'to play:Execute play('''||sym2||''',x,y);');

else

dbms\_output.enable(10000);

dbms\_output.put\_line('You cant play this block it is already filled');

end if;

end if;

end;

/

execute print\_gameboard;

execute reset\_gameboard;

execute playgame('X',1,1);

execute playgame('O',2,2);

execute playgame('X',3,1);

execute playgame('O',1,1);

create or replace PROCEDURE PlayerMovettt(p\_move IN VARCHAR2, p\_column IN VARCHAR2, p\_row IN NUMBER)

IS

BEGIN

-- Check for valid sign

IF p\_move NOT IN ('X', 'O')

THEN dbms\_output.put\_line('Move must be X or O');

END IF;

-- Check for valid col

IF p\_column NOT IN ('C1', 'C2', 'C3')

THEN dbms\_output.put\_line('Column must be C1, C2 or C3');

END IF;

-- Check for valid row

IF p\_row NOT IN (1,2,3)

THEN dbms\_output.put\_line('Row must be 1, 2 or 3');

END IF;

-- SQLINES DEMO \*\*\* turn and update player turn

IF p\_move = (SELECT turn FROM ttt\_PlayerTurn2)

THEN (SELECT CONCAT('This turn belongs to player ', (SELECT turn FROM ttt\_PlayerTurn2), '!') FROM Dual);

ELSE

UPDATE TicTacToeGame

SET p\_column = p\_move

WHERE row\_num = p\_row;

UPDATE ttt\_PlayerTurn2

SET turn =

CASE

WHEN turn = 'X' THEN 'O'

WHEN turn = 'O' THEN 'X'

END;

END IF;

/\*CheckVictoryttt();\*/

END PlayerMovettt;

/\*

create or replace PROCEDURE CheckVictoryttt

IS

A1 VARCHAR2(1);

A2 VARCHAR2(1);

A3 VARCHAR2(1);

B1 VARCHAR2(1);

B2 VARCHAR2(1);

B3 VARCHAR2(1);

C1 VARCHAR2(1);

C2 VARCHAR2(1);

C3 VARCHAR2(1);

BEGIN

SELECT A INTO A1 FROM TicTacToe WHERE ID = 1;

SELECT A INTO A2 FROM TicTacToe WHERE ID = 2;

SELECT A INTO A3 FROM TicTacToe WHERE ID = 3;

SELECT B INTO B1 FROM TicTacToe WHERE ID = 1;

SELECT B INTO B2 FROM TicTacToe WHERE ID = 2;

SELECT B INTO B3 FROM TicTacToe WHERE ID = 3;

SELECT C INTO C1 FROM TicTacToe WHERE ID = 1;

SELECT C INTO C2 FROM TicTacToe WHERE ID = 2;

SELECT C INTO C3 FROM TicTacToe WHERE ID = 3;

CASE

-- Horizontal wins

WHEN

A1 = B1 AND B1 = C1

THEN (SELECT \*, "Player " || A1 || ' is victorious!' AS "Result" FROM TicTacToe);

WHEN

A2 = B2 AND B2 = C2

THEN (SELECT \*, "Player " || A2 || ' is victorious!' AS "Result" FROM TicTacToe);

WHEN

A3 = B3 AND B3 = C3

THEN (SELECT \*, "Player " || A3 || ' is victorious!' AS "Result" FROM TicTacToe);

-- Vertical wins

WHEN

A1 = A2 AND A2 = A3

THEN (SELECT \*, "Player " || A1 || ' is victorious!' AS "Result" FROM TicTacToe);

WHEN

B1 = B2 AND B2 = B3

THEN (SELECT \*, "Player " || B1 || ' is victorious!' AS "Result" FROM TicTacToe);

WHEN

C1 = C2 AND C2 = C3

THEN (SELECT \*, "Player " || C1 || ' is victorious!' AS "Result" FROM TicTacToe);

-- Diagonal wins

WHEN

A1 = B2 AND B2 = C3

THEN (SELECT \*, "Player " || A1 || ' is victorious!' AS "Result" FROM TicTacToe);

WHEN

A3 = B2 AND B2 = C1

THEN (SELECT \*, "Player " || A3 || ' is victorious!' AS "Result" FROM TicTacToe);

-- Game continues

ELSE (SELECT \*, 'Game is still ongoing' AS "Result" FROM TicTacToe);

END CASE;

END;\*/