ASSIGNMENT 3: TICTACTOE GAME

- Code is given below as well as it is uploaded in GitHub link. Attaching SQL script as well.
- Test cases with script output obtained on console are mentioned after the code.
- Please run below scripts one after the other in given order to generate required tables, function, and stored procedures.

--CREATE TABLE TICTACTOE

```
create table tictactoe (id number, A varchar2(1), B varchar2(1), C varchar2(1));
```

--INSERT VALUES INTO TICTACTOE TABLE AS GIVEN BELOW

```
insert into tictactoe values(1, ' ', ' ', ' ');

insert into tictactoe values(2, ' ', ' ', ' ');

insert into tictactoe values(3, ' ', ' ', ' ');

select * from tictactoe;
```

--CREATE TABLE TRACKER : This table is for tracking all the positions that are open and the respective values currently entered at those positions

```
create table tracker (row_val number, col_val number, occupied number, entry_value number);
```

--INSERT DATA INTO TRACKER TABLE

--In this table 0 or 1 values entered in 'occupied' field specifies whether that field is occupied or not.

--The entry value field can have 3 values null, 1 and 2 where value 1 denotes entry 'X' and value 2 denotes entry 'O'

```
insert into tracker values(1, 1, 0, null);

insert into tracker values(1, 2, 0, null);

insert into tracker values(1, 3, 0, null);

insert into tracker values(2, 1, 0, null);

insert into tracker values(2, 2, 0, null);

insert into tracker values(2, 3, 0, null);
```

```
insert into tracker values(3, 1, 0, null);

insert into tracker values(3, 2, 0, null);

insert into tracker values(3, 3, 0, null);

select * from tracker;
```

```
--CREATE TABLE PLAYER

--This table will have 2 fields id, player_id

--This table is to keep a track of which player's turn it is.

create table player(id number, player_id number(1));
```

```
--INSERT VALUE INTO PLAYER TABLE

--This table will have id as 1 and player_id as 1 at the start of the game.

--player_id specifies the player which will play. Game starts with player 1 as player_id is 1 at the start of the game.

insert into player values(1,1);

select * from player;
```

```
--CREATE STORED PROCEDURE viewBoard

--This stored procedure is used to view where the players have entered 'X' and 'O'

CREATE OR REPLACE PROCEDURE viewBoard

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;

dbms_output.put_line('|' || a1 || '|' || b1 || '|' || c1 || '|');
dbms_output.put_line('|' || a2 || '|' || b2 || '|' || c2 || '|');
dbms_output.put_line('|' || a3 || '|' || b3 || '|' || c3 || '|');

END viewBoard;
```

```
--CREATE FUNCTION CHECKWIN
--This function checks if there is a win condition and returns a flag accordingly.
CREATE OR REPLACE FUNCTION checkWin
RETURN boolean IS
  win_flag boolean;
  a1 number;
  a2 number;
  a3 number;
  b1 number;
  b2 number;
  b3 number;
  c1 number;
```

```
  c2 number;
  c3 number;
BEGIN
   select entry_value into a1 from tracker where row_val=1 and col_val=1;
   select entry_value into a2 from tracker where row_val=2 and col_val=1;
   select entry_value into a3 from tracker where row_val=3 and col_val=1;
   select entry_value into b1 from tracker where row_val=1 and col_val=2;
   select entry_value into b2 from tracker where row_val=2 and col_val=2;
   select entry_value into b3 from tracker where row_val=3 and col_val=2;
   select entry_value into c1 from tracker where row_val=1 and col_val=3;
   select entry_value into c2 from tracker where row_val=2 and col_val=3;
   select entry_value into c3 from tracker where row_val=3 and col_val=3;

   IF ((a1=a2 and a2=a3) or (b1=b2 and b2=b3) or (c1=c2 and c2=c3) or (a1=b1 and b1=c1) or (a2=b2 and
b2=c2) or (a3=b3 and b3=c3) or (a1=b2 and b2=c3) or (c1=b2 and b2=a3)) THEN
      win_flag:=True;

   ELSE
      win_flag:=False;

   END IF;
   RETURN win_flag;
END;
```

```
--CREATE STORED PROCEDURE PLAYGAME
CREATE OR REPLACE PROCEDURE PLAYGAME (col_value IN NUMBER, row_value IN NUMBER)
AS
occupancy NUMBER;
player_id NUMBER;
entry_count NUMBER;
```

```
BEGIN
    select count(occupied) into entry_count from tracker where occupied=1;
    IF entry_count=9 THEN
      DBMS_OUTPUT.PUT_LINE('GAME IS A DRAW!');
      viewBoard;
      update tracker set occupied=0;
      update tracker set entry_value=null;
      update tictactoe set A=' ';
      update tictactoe set B=' ';
      update tictactoe set C=' ';
      update player set player_id=1 where id=1;
    ELSE
      select player_id into player_id from player where id=1;
      select occupied into occupancy from tracker where row_val=row_value and col_val=col_value;
      IF occupancy=0 THEN
        DBMS_OUTPUT.PUT_LINE('AVAILABLE POSITION');
        IF col_value=1 THEN
          update tracker set occupied=1 where row_val=row_value and col_val=col_value;
          IF player_id=1 THEN
            update tictactoe set A='X' where id = row_value;
            update tracker set entry_value=1 where row_val=row_value and col_val=col_value;
          ELSE
            update tictactoe set A='O' where id = row_value;
            update tracker set entry_value=2 where row_val=row_value and col_val=col_value;
          END IF;
        ELSIF col_value=2 THEN
          update tracker set occupied=1 where row_val=row_value and col_val=col_value;
          IF player_id=1 THEN
            update tictactoe set B='X' where id = row_value;
            update tracker set entry_value=1 where row_val=row_value and col_val=col_value;
```

```
    ELSE
        update tictactoe set B='O' where id = row_value;
        update tracker set entry_value=2 where row_val=row_value and col_val=col_value;
    END IF;
ELSIF col_value=3 THEN
    update tracker set occupied=1 where row_val=row_value and col_val=col_value;
    IF player_id=1 THEN
        update tictactoe set C='X' where id = row_value;
        update tracker set entry_value=1 where row_val=row_value and col_val=col_value;
    ELSE
        update tictactoe set C='O' where id = row_value;
        update tracker set entry_value=2 where row_val=row_value and col_val=col_value;
    END IF;
END IF;

IF checkWin THEN
    viewBoard;
    update tracker set occupied=0;
    update tracker set entry_value=null;
    update tictactoe set A=' ';
    update tictactoe set B=' ';
    update tictactoe set C=' ';
    update player set player_id=1 where id=1;
    DBMS_OUTPUT.PUT_LINE('PLAYER ' || player_id || ' WINS!!');
ELSE
    viewBoard;
    select count(occupied) into entry_count from tracker where occupied=1;
    IF entry_count=9 THEN
        DBMS_OUTPUT.PUT_LINE('GAME IS A DRAW!');
        update tracker set occupied=0;
```

```
            update tracker set entry_value=null;
            update tictactoe set A=' ';
            update tictactoe set B=' ';
            update tictactoe set C=' ';
            update player set player_id=1 where id=1;
          ELSE
            IF player_id=1 THEN
              update player set player_id=2 where id=1;
              DBMS_OUTPUT.PUT_LINE('********PLAYER 2 TURN NEXT********');
            ELSIF player_id=2 THEN
              update player set player_id=1 where id=1;
              DBMS_OUTPUT.PUT_LINE('********PLAYER 1 TURN NEXT********');
            END IF;
          END IF;
        END IF;
      ELSE
        DBMS_OUTPUT.PUT_LINE('POSITION TAKEN. TURN GONE');
        viewBoard;
        IF player_id=1 THEN
          update player set player_id=2 where id=1;
          DBMS_OUTPUT.PUT_LINE('********PLAYER 2 TURN NEXT********');
        ELSIF player_id=2 THEN
          update player set player_id=1 where id=1;
          DBMS_OUTPUT.PUT_LINE('********PLAYER 1 TURN NEXT********');
        END IF;
      END IF;
    END IF;
END PLAYGAME;
```

--GAME CONDITIONS & FLOW OF CODE

--The game starts with the 'player_id' which is present in the 'player' table.

--The 'player_id' is set to 1 by default in the 'player' table(before the game and at the end of every game).

--Player 1 will have entry 'X' on the board and player 2 will have entry 'O' on the board.

--Game starts with player 1 where the player enters the column and row value in the stored procedure 'PLAYGAME'

--Once column and row positions are taken from the player, the 'PLAYGAME' stored procedure executes.

--The 'PLAYGAME' procedure makes use of the 'tracker' table to keep a track of occupied/unoccupied positions and inserts value in 'tictactoe' table accordingly.

--Every move each player makes is tracked in the 'tracker' table using 'occupied' and 'entry_value' fields.

--The tictactoe board is displayed after every turn and whose turn it is next is also notified in the console.

--The player is automatically switched to the next player in player table through the stored procedure 'PLAYGAME'. Thus, the game continues after every player makes their move till there is a win or draw condition.

--If the player enters a position which is already occupied, then they get a message 'POSITION TAKEN. TURN GONE' and the turn goes to the other player.

--The game ends when the game is a DRAW or a WIN condition and the tictactoe board (and the tictactoe, tracker, player tables) is reset.

--Winning player is also displayed on the console. Also, message regarding DRAW is printed.

--The game can be played any number of times. The game must be completed(WIN or DRAW condition must be achieved) before starting a new game.

```
--TEST CASES
--INSERT COLUMN POSITION FIRST AND THEN ROW POSITION IN PLAYGAME STORED PROCEDURE.
--TEST CASE FOR HORIZONTAL WIN
SET SERVEROUTPUT ON;
--Player 1:
EXECUTE PLAYGAME(3,1);

--Player 2:
EXECUTE PLAYGAME(2,3);

--Player 1:
EXECUTE PLAYGAME(2,2);

--Player 2:
EXECUTE PLAYGAME(1,3);

--Player 1:(Below condition should give 'POSITION TAKE. TURN GONE MESSAGE')
EXECUTE PLAYGAME(1,3);

--Player 2:
EXECUTE PLAYGAME(3,3);
```

```
AVAILABLE POSITION
| | |X|
| | | |
| | | |
********PLAYER 2 TURN NEXT********


PL/SQL procedure successfully completed.

AVAILABLE POSITION
| | |X|
| | | |
| |O| |
********PLAYER 1 TURN NEXT********


PL/SQL procedure successfully completed.

AVAILABLE POSITION
| | |X|
| |X| |
| |O| |
********PLAYER 2 TURN NEXT********


PL/SQL procedure successfully completed.

AVAILABLE POSITION
| | |X|
| |X| |
|O|O| |
********PLAYER 1 TURN NEXT********


PL/SQL procedure successfully completed.
```

```
POSITION TAKEN. TURN GONE
| | |X|
| |X| |
|O|O| |
********PLAYER 2 TURN NEXT********


PL/SQL procedure successfully completed.

AVAILABLE POSITION
| | |X|
| |X| |
|O|O|O|
PLAYER 2 WINS!!


PL/SQL procedure successfully completed.
```

```
--TEST CASE 2 : VERTICAL WIN CONDITION

SET SERVEROUTPUT ON;

--Player 1:

EXECUTE PLAYGAME(1,1);

--Player 2:

EXECUTE PLAYGAME(2,1);

--Player 1:

EXECUTE PLAYGAME(2,2);

--Player 2: (Below conditon should give 'POSITION TAKE. TURN GONE MESSAGE')

EXECUTE PLAYGAME(2,2);

--Player 1:

EXECUTE PLAYGAME(1,2);

--Player 2:

EXECUTE PLAYGAME(3,2);

--Player 1:

EXECUTE PLAYGAME(1,3);
```

```
AVAILABLE POSITION
|X| | |
| | | |
| | | |
********PLAYER 2 TURN NEXT********


PL/SQL procedure successfully completed.

AVAILABLE POSITION
|X|O| |
| | | |
| | | |
********PLAYER 1 TURN NEXT********


PL/SQL procedure successfully completed.

AVAILABLE POSITION
|X|O| |
| |X| |
| | | |
********PLAYER 2 TURN NEXT********


PL/SQL procedure successfully completed.

POSITION TAKEN. TURN GONE
|X|O| |
| |X| |
| | | |
********PLAYER 1 TURN NEXT********


PL/SQL procedure successfully completed.
```

```
AVAILABLE POSITION
|X|O| |
|X|X| |
| | | |
********PLAYER 2 TURN NEXT********


PL/SQL procedure successfully completed.

AVAILABLE POSITION
|X|O| |
|X|X|O|
| | | |
********PLAYER 1 TURN NEXT********


PL/SQL procedure successfully completed.

AVAILABLE POSITION
|X|O| |
|X|X|O|
|X| | |
PLAYER 1 WINS!!


PL/SQL procedure successfully completed.
```

```
--TEST CASE 3: DIAGONAL WIN CONDITION

SET SERVEROUTPUT ON;

--Player 1:

EXECUTE PLAYGAME(2,1);

--Player 2:

EXECUTE PLAYGAME(1,2);

--Player 1:

EXECUTE PLAYGAME(3,1);

--Player 2:

EXECUTE PLAYGAME(1,1);

--Player 1:

EXECUTE PLAYGAME(1,3);

--Player 2:

EXECUTE PLAYGAME(2,2);

--Player 1:

EXECUTE PLAYGAME(3,2);

--Player 2:

EXECUTE PLAYGAME(3,3);
```

```
AVAILABLE POSITION
| |X| |
| | | |
| | | |
********PLAYER 2 TURN NEXT********


PL/SQL procedure successfully completed.

AVAILABLE POSITION
| |X| |
|O| | |
| | | |
********PLAYER 1 TURN NEXT********


PL/SQL procedure successfully completed.

AVAILABLE POSITION
| |X|X|
|O| | |
| | | |
********PLAYER 2 TURN NEXT********


PL/SQL procedure successfully completed.

AVAILABLE POSITION
|O|X|X|
|O| | |
| | | |
********PLAYER 1 TURN NEXT********


PL/SQL procedure successfully completed.
```

```
AVAILABLE POSITION
|O|X|X|
|O| | |
|X| | |
********PLAYER 2 TURN NEXT********


PL/SQL procedure successfully completed.

AVAILABLE POSITION
|O|X|X|
|O|O| |
|X| | |
********PLAYER 1 TURN NEXT********


PL/SQL procedure successfully completed.

AVAILABLE POSITION
|O|X|X|
|O|O|X|
|X| | |
********PLAYER 2 TURN NEXT********


PL/SQL procedure successfully completed.

AVAILABLE POSITION
|O|X|X|
|O|O|X|
|X| |O|
PLAYER 2 WINS!!
```

```
PL/SQL procedure successfully completed.
```

```
--TEST CASE 4: GAME DRAW CONDITION

SET SERVEROUTPUT ON;
--Player 1:
EXECUTE PLAYGAME(2,1);

--Player 2:
EXECUTE PLAYGAME(2,2);

--Player 1:
EXECUTE PLAYGAME(2,3);

--Player 2:
EXECUTE PLAYGAME(3,1);

--Player 1:
EXECUTE PLAYGAME(1,3);

--Player 2:
EXECUTE PLAYGAME(3,3);

--Player 1:
EXECUTE PLAYGAME(3,2);

--Player 2:
EXECUTE PLAYGAME(1,2);

--Player 1:
EXECUTE PLAYGAME(1,1);
```

```
AVAILABLE POSITION
| |X| |
| | | |
| | | |
********PLAYER 2 TURN NEXT********


PL/SQL procedure successfully completed.

AVAILABLE POSITION
| |X| |
| |O| |
| | | |
********PLAYER 1 TURN NEXT********


PL/SQL procedure successfully completed.

AVAILABLE POSITION
| |X| |
| |O| |
| |X| |
********PLAYER 2 TURN NEXT********


PL/SQL procedure successfully completed.

AVAILABLE POSITION
| |X|O|
| |O| |
| |X| |
********PLAYER 1 TURN NEXT********


PL/SQL procedure successfully completed.
```

```
AVAILABLE POSITION
| |X|O|
| |O| |
|X|X| |
********PLAYER 2 TURN NEXT********


PL/SQL procedure successfully completed.

AVAILABLE POSITION
| |X|O|
| |O| |
|X|X|O|
********PLAYER 1 TURN NEXT********


PL/SQL procedure successfully completed.

AVAILABLE POSITION
| |X|O|
| |O|X|
|X|X|O|
********PLAYER 2 TURN NEXT********


PL/SQL procedure successfully completed.

AVAILABLE POSITION
| |X|O|
|O|O|X|
|X|X|O|
********PLAYER 1 TURN NEXT********
```

```
PL/SQL procedure successfully completed.

AVAILABLE POSITION
|X|X|O|
|O|O|X|
|X|X|O|
GAME IS A DRAW!


PL/SQL procedure successfully completed.
```

```
--DELETE SCRIPTS
drop function checkWin;

drop procedure viewBoard;

drop procedure playGame;

drop procedure viewBoard;

drop table player;

drop table tracker;

drop table tictactoe;
```