Assignment 3: Haarika Sai Katlaparthi

```
set serveroutput on;

-- Creation of the Assignement3 table if it does not exist
DECLARE
  num1 NUMBER;
BEGIN
  SELECT count(*) INTO num1 FROM user_tables
    WHERE TABLE_NAME = 'Assignement3';
  IF num1 = 0 THEN
    EXECUTE IMMEDIATE 'CREATE TABLE Assignement3(
      y NUMBER,
      A CHAR,
      B CHAR,
      C CHAR
    )';
 END IF;
END;
/

-- Number to column name conversion function
CREATE OR REPLACE FUNCTION ColName(n1 IN NUMBER)
RETURN CHAR
IS
BEGIN
  IF n1=1 THEN
    RETURN 'A';
  ELSIF n1=2 THEN
    RETURN 'B';
  ELSIF n1=3 THEN
    RETURN 'C';
  ELSE
    RETURN '_';
  END IF;
END;
/

-- Procedure to display the game board
CREATE OR REPLACE PROCEDURE show_game IS
BEGIN
  dbms_output.enable(1000000);
  dbms_output.put_line(' ');
  FOR ll in (SELECT * FROM Assignement3 ORDER BY Y) LOOP
    dbms_output.put_line('     ' || ll.A || ' ' || ll.B || ' ' || ll.C);
  END LOOP;
  dbms_output.put_line(' ');
END;
```

```
/
-- Game reset procedure
CREATE OR REPLACE PROCEDURE reset_game IS
a1 NUMBER;
BEGIN
  DELETE FROM Assignement3;
  FOR a1 in 1..3 LOOP
    INSERT INTO Assignement3 VALUES (a1,'_','_','_');
  END LOOP;
  dbms_output.enable(1000000);
  show_game();
  dbms_output.put_line('The game is ready to play : EXECUTE play(''X'', x, y);');
END;
/

-- Procedure to play
CREATE OR REPLACE PROCEDURE play(s1 IN VARCHAR2, colnum IN NUMBER, rownumm IN NUMBER) IS
val Assignement3.a%type;
col1 CHAR;
s2 CHAR;
BEGIN
  SELECT ColName(colnum) INTO col1 FROM DUAL;
  EXECUTE IMMEDIATE ('SELECT ' || col1 || ' FROM Assignement3 WHERE y=' || rownumm) INTO val;
IF val!='_' THEN
    dbms_output.enable(1000000);
    dbms_output.put_line('Its a tie');
    END IF;
  IF val='_' THEN
    EXECUTE IMMEDIATE ('UPDATE Assignement3 SET ' || col1 || '=''' || s1 || ''' WHERE y=' ||
rownumm);
    IF s1='X' THEN
      s2:='O';
    ELSE
      s2:='X';
    END IF;
    show_game();
    dbms_output.put_line('Around ' || s2 || '. to play : EXECUTE play(''' || s2 || ''', x, y);');
  ELSE
    dbms_output.enable(1000000);
    dbms_output.put_line('You cannot play this square, it is already played');
  END IF;
END;
/
```

-- Procedure to declare win

```
CREATE OR REPLACE PROCEDURE winner(s1 IN VARCHAR2) IS
BEGIN
  dbms_output.enable(1000000);
  show_game();
  dbms_output.put_line('The player ' || s1 || ' won !!');
  dbms_output.put_line('---------------------------------------');
  dbms_output.put_line('Starting a new game...');
  reset_game();
END;
/
```

-- Column query creation function

```
CREATE OR REPLACE FUNCTION wincol(numcol IN VARCHAR2, s1 IN VARCHAR2)
RETURN VARCHAR2
IS
BEGIN
  RETURN ('SELECT COUNT(*) FROM Assignement3 WHERE ' || numcol || ' = '''|| s1 ||''' AND ' || numcol
|| ' != ''_''');
END;
/
```

-- Column query creation function

```
CREATE OR REPLACE FUNCTION wincross(numcol IN VARCHAR2, yvalue IN NUMBER)
RETURN VARCHAR2
IS
BEGIN
  RETURN ('SELECT '|| numcol ||' FROM Assignement3 WHERE y=' || yvalue);
END;
/
```

-- column test function

```
CREATE OR REPLACE FUNCTION wincoll(numcol IN VARCHAR2)
RETURN CHAR
IS
  winnum NUMBER;
  r VARCHAR2(100);
BEGIN
  SELECT wincol(numcol, 'X') into r FROM DUAL;
  EXECUTE IMMEDIATE r INTO winnum;
  IF winnum=3 THEN
    RETURN 'X';
  ELSIF winnum=0 THEN
```

```
   SELECT wincol(numcol, 'O') into r FROM DUAL;
   EXECUTE IMMEDIATE r INTO winnum;
   IF winnum=3 THEN
     RETURN 'O';
   END IF;
  END IF;
  RETURN '_';
END;
/
```

-- Diagonal test function

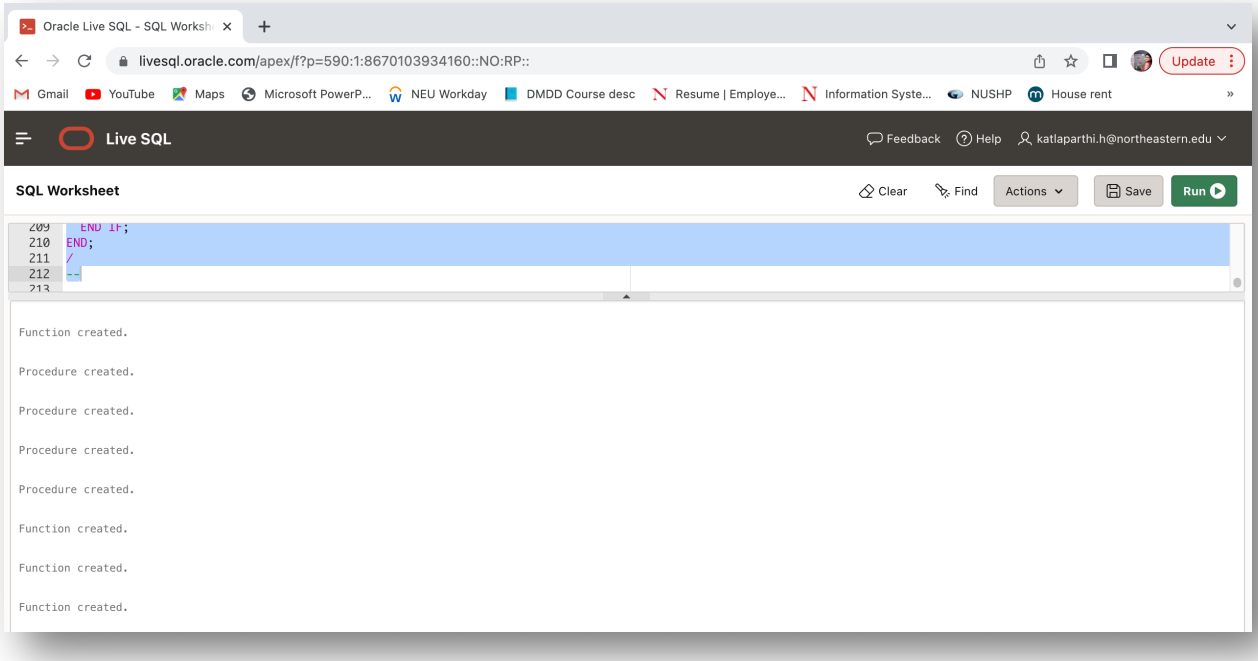
```
CREATE OR REPLACE FUNCTION wincross1(tmpx IN CHAR, numcol IN NUMBER, numrow IN NUMBER)
RETURN CHAR
IS
  tmpvar CHAR;
  tmpxvar CHAR;
  r VARCHAR2(100);
BEGIN
  SELECT wincross(ColName(numcol), numrow) INTO r FROM DUAL;
  IF tmpx IS NULL THEN
    EXECUTE IMMEDIATE (r) INTO tmpxvar;
  ELSIF NOT tmpx = '_' THEN
    EXECUTE IMMEDIATE (r) INTO tmpvar;
    IF NOT tmpx = tmpvar THEN
      tmpxvar := '_';
    END IF;
  ELSE
    tmpxvar := '_';
  END IF;
  RETURN tmpxvar;
END;
/
```

-- Test trigger if we win

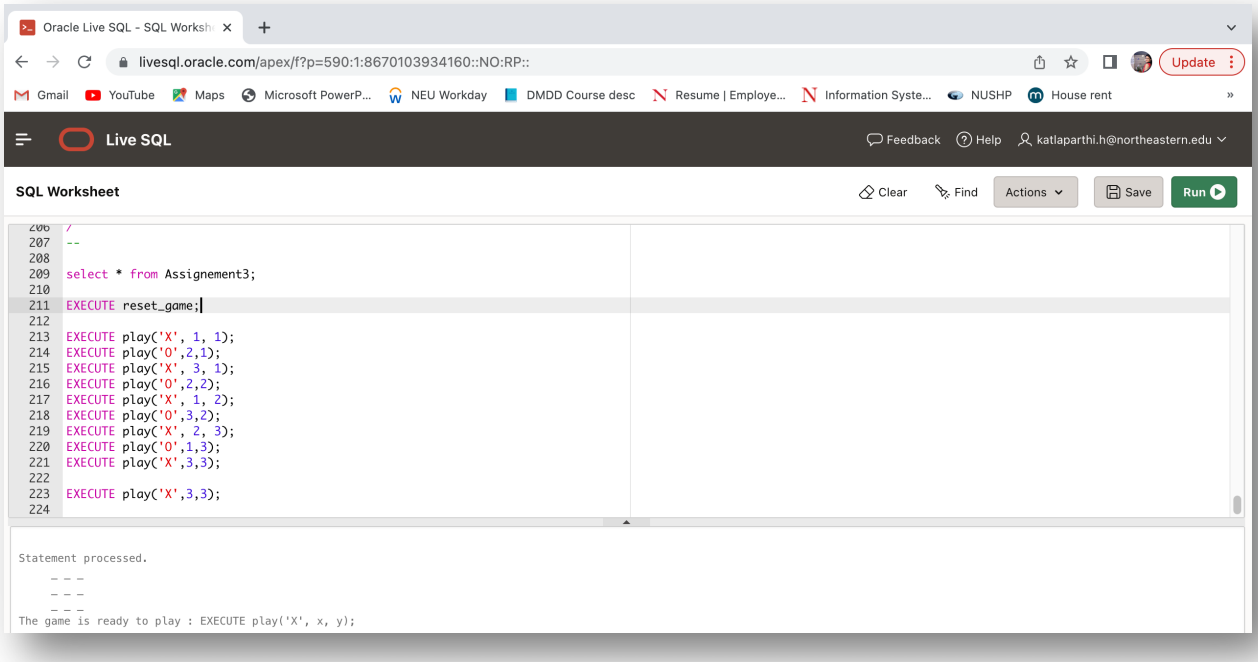
```
CREATE OR REPLACE TRIGGER iswinner1
AFTER UPDATE ON Assignement3
DECLARE
  CURSOR cr_ligne IS
    SELECT * FROM Assignement3 ORDER BY Y;
  crlv Assignement3%rowtype;
  tmpvar CHAR;
  tmpx1 CHAR;
  tmpx2 CHAR;
  r VARCHAR2(40);
BEGIN
  FOR crlv IN cr_ligne LOOP
    -- line test
    IF crlv.A = crlv.B AND crlv.B = crlv.C AND NOT crlv.A='_' THEN
```

```
    winner(crlv.A);
    EXIT;
   END IF;
   -- column test
   SELECT wincoll(ColName(crlv.Y)) INTO tmpvar FROM DUAL;
   IF NOT tmpvar = '_' THEN
    winner(tmpvar);
    EXIT;
   END IF;
   -- diagonal test
   SELECT wincross1(tmpx1, crlv.Y, crlv.Y) INTO tmpx1 FROM dual;
   SELECT wincross1(tmpx2, 4-crlv.Y, crlv.Y) INTO tmpx2 FROM dual;
  END LOOP;
  IF NOT tmpx1 = '_' THEN
   winner(tmpx1);
  END IF;
  IF NOT tmpx2 = '_' THEN
   winner(tmpx2);
  END IF;
END;
/
--
```

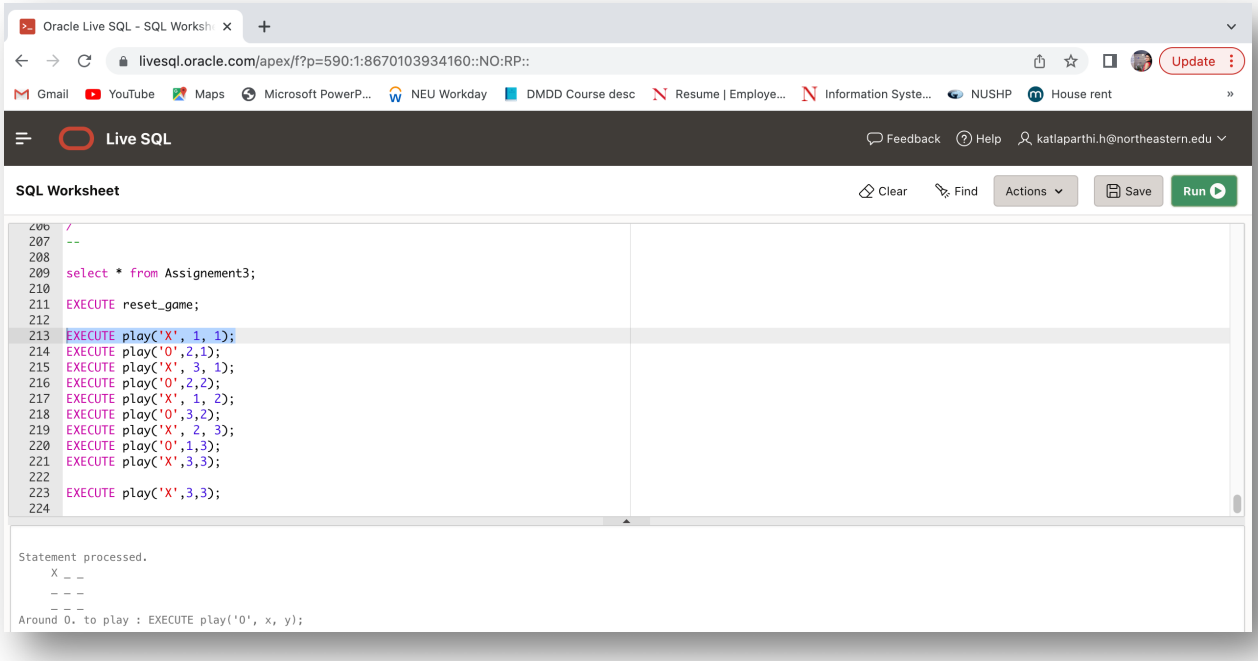
Executed all procedures.



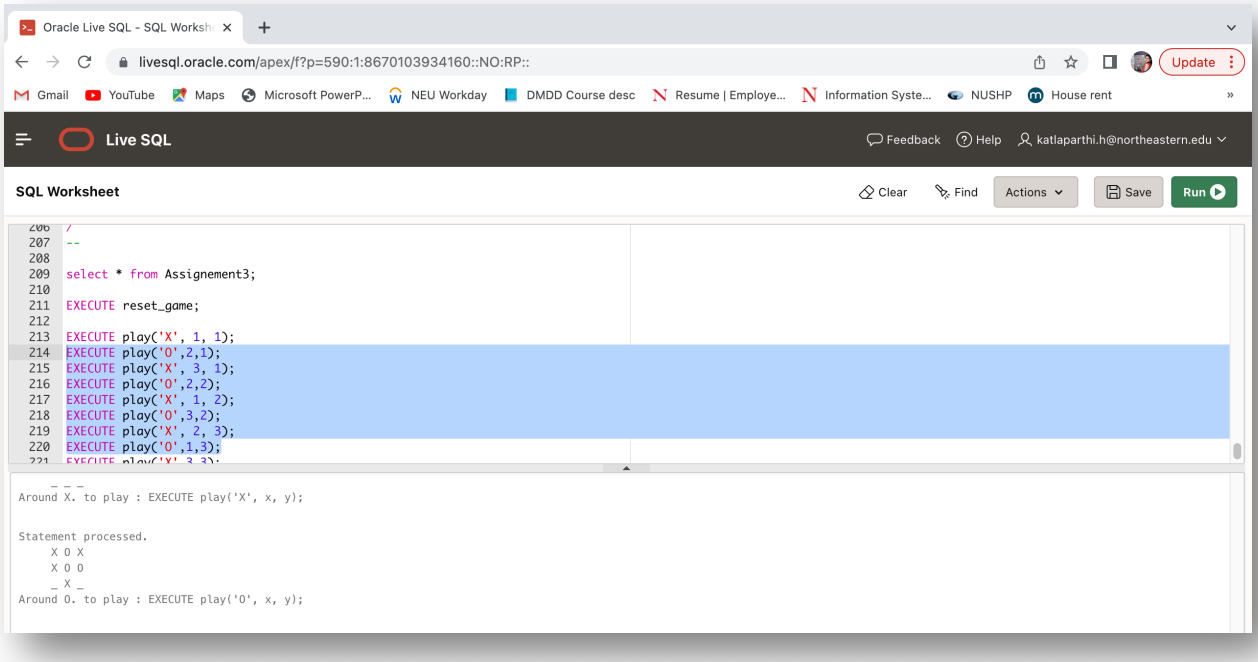
Reset game



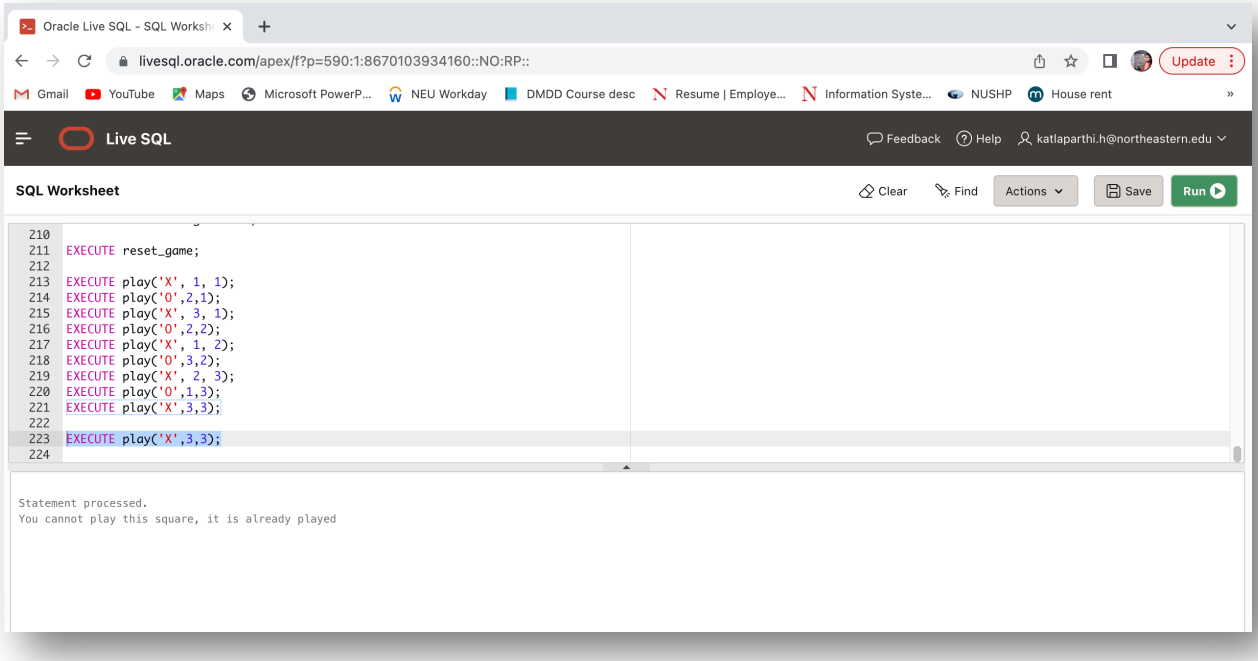
Case 1: Started Game



Case 2: Game on

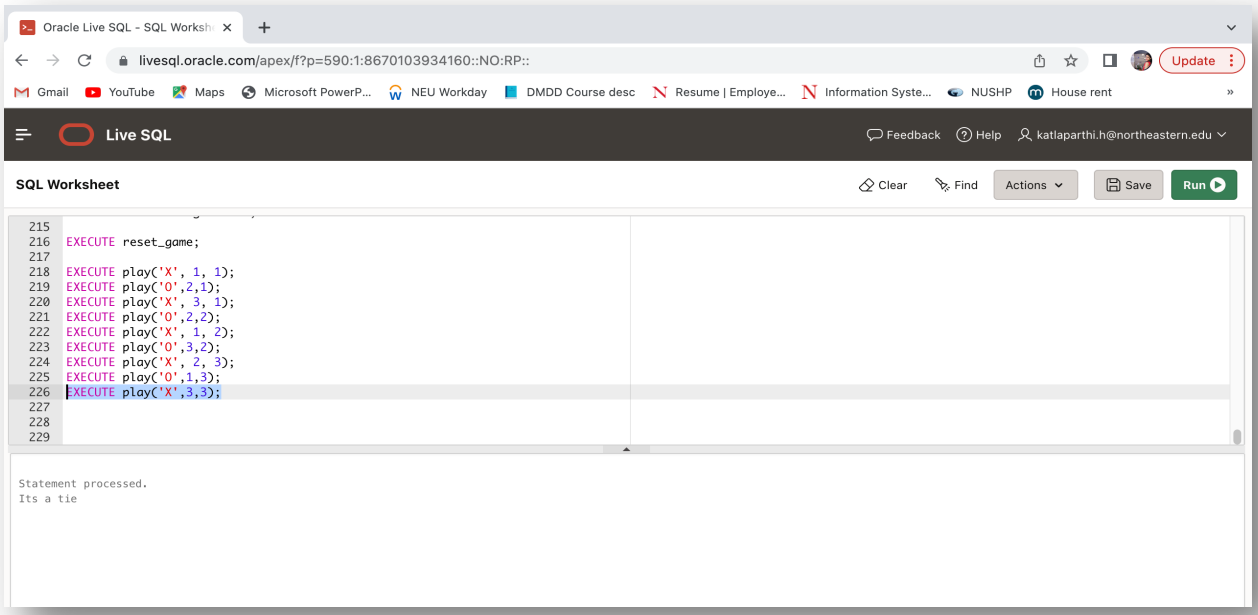


Case 3: Cannot reach



Case 4: Tie

```
EXECUTE play('X', 1, 1);
EXECUTE play('O',2,1);
EXECUTE play('X', 3, 1);
EXECUTE play('O',2,2);
EXECUTE play('X', 1, 2);
EXECUTE play('O',3,2);
EXECUTE play('X', 2, 3);
EXECUTE play('O',1,3);
EXECUTE play('X',3,3);
```



Case 5: One of the player Won

EXECUTE play('X', 1, 1);
EXECUTE play('O',2,1);
EXECUTE play('X', 3, 1);
EXECUTE play('O',2,2);
EXECUTE play('X', 1, 2);
EXECUTE play('O',2,3);