



CFGS DESENVOLUPAMENT D'APLICACIONS MULTIPLATAFORMA - LOGÍSTICA

MÒDUL 02: Bases de dades

NOM I COGNOMS:

Nota:

UF 3: DCL i Rutines.

CURS: 22 - 23

Grup DAM1T

PRÀCTICA 07: RUTINES

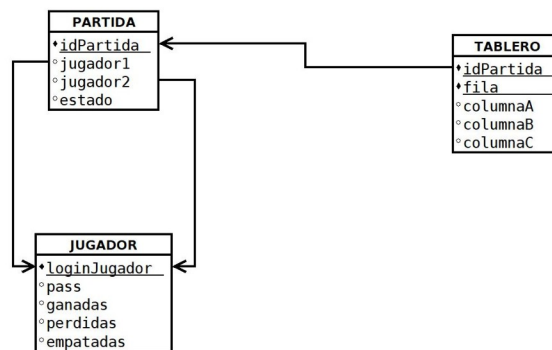
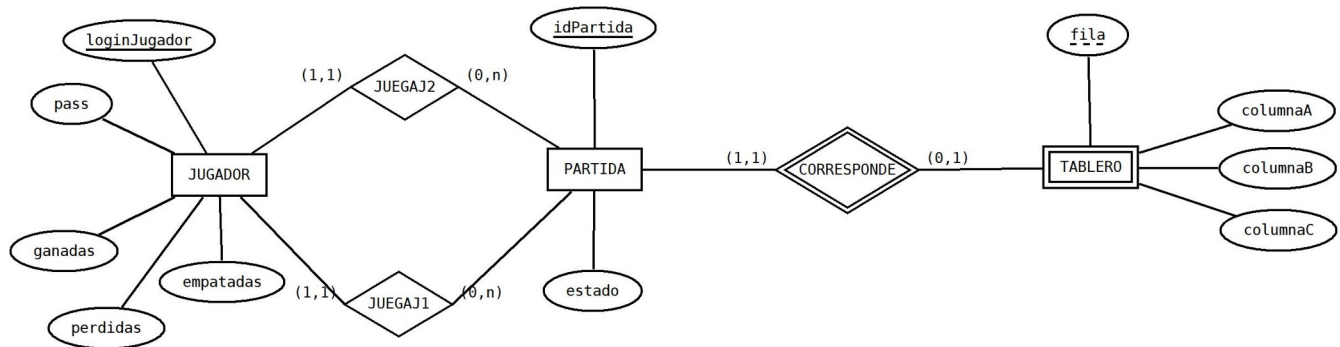
Tres en raya

El tres en raya es un juego de tablero entre dos jugadores, que alternadamente marcan las casillas de un tablero de 3x3 con sus símbolos ('X' o 'O'). El objetivo de ser el primero en alinear tres de sus símbolos en línea recta (horitzontal, vertical o diagonal).

Los jugadores no tardan en descubrir que el juego perfecto termina en empate sin importar con qué juega el primer jugador. La misma simplicidad del juego de tres en raya lo hacen ideal como herramienta pedagógica para enseñar los conceptos de teoría de juegos y la rama de inteligencia artificial.

Sus orígenes se pierden en la antigüedad. Hay documentados juegos de tres en raya en el antiguo Egipto (3000 aC), Creta, Grecia y Roma. Posteriormente se han hallado muestras en China (500 dC) y Sri Lanka.

<https://elpais.com/cultura/2022-03-26/los-soldados-romanos-mataban-el-tiempo-con-juegos-estrategicos-de-mesa.html>



CFGS DESENVOLUPAMENT D'APLICACIONS MULTIPLATAFORMA - LOGÍSTICA			
MÒDUL 02: Bases de dades			
NOM I COGNOMS:			Nota:
UF 3: DCL i Rutines.	CURS: 22 - 23	Grup DAMIT	

P1) (3 puntos) 3enRaya FUNCTION:

```
# Function tablero_ganador(idPartidaP int) returns varchar(50)
# (3 puntos)
# Generar una funcion que consultando las casillas marcadas de la tabla TABLERO averigüe si el tablero de la partida especificada como
parametro tiene ganador (hay un tres en raya) y por tanto retorna 'GANADOR JUGADOR1' (si son tres 'X'), o 'GANADOR JUGADOR2' (si
son tres 'O'), o retorna 'FALSO' en caso contrario.
#

# TEST tablero_ganador
call mostrar_tablero(1);
select tablero_ganador(1); # FALSO
call mostrar_tablero(2);
select tablero_ganador(2); # GANADOR JUGADOR1
call mostrar_tablero(3);
select tablero_ganador(3); # FALSO
```


P2) (4 puntos) 3enRaya TRIGGER:

```
# Trigger modificar_estado after update on TABLERO for each row
# (4 puntos)
# Generar un disparador que cuando se modifica el tablero (porque un jugador hace un movimiento), llame a las funciones tablero_ganador
y tablero_lleno pasandoles el identificador de partida sobre el que se hace el update, y entonces evalua la informacion que retornan y
decide:
# Si la funcion tablero_ganador retorna 'GANADOR JUGADOR1' o 'GANADOR JUGADOR2', es que en el tablero hay un ganador. Entonces
debe modificar el estado en la tabla PARTIDA con el valor que corresponda de 'GANADOR JUGADOR1' o 'GANADOR JUGADOR2', y
despues modificar en la tabla JUGADOR los jugadores de esa partida, incrementando el campo ganadas del ganador, y el campo perdidas
del perdedor.
# Si no hay un ganador, y la funcion tablero_lleno retorna 'CIERTO', es que el tablero esta lleno, y no quedan casillas vacias para que los
jugadores sigan haciendo movimientos. Entonces debe modificar el estado en la tabla PARTIDA con el valor 'EMPATE', y despues modificar
en la tabla JUGADOR los jugadores de esa partida, incrementando el campo empatadas de ambos.
#

# TEST modificar_estado
# Partida 4 estado 'TURNO JUGADOR1', si mueve en A3 futuro 'EMPATE'
call mostrar_tablero(4);
select * from JUGADOR;
select * from PARTIDA;
update TABLERO set columnaA='X' where idPartida = 4 and fila = 3;
call mostrar_tablero(4);
select * from JUGADOR; # Jugadores rojo y azul incrementan empatadas
select * from PARTIDA; # Estado de partida 4 pasa a 'EMPATE'

# Partida 5 estado 'TURNO JUGADOR2', si mueve en A2 futuro 'GANADOR JUGADOR2'
call mostrar_tablero(5);
select * from JUGADOR;
select * from PARTIDA;
update TABLERO set columnaA='O' where idPartida = 5 and fila = 2;
call mostrar_tablero(5);
select * from JUGADOR; # Jugador azul incrementa ganadas y jugador rojo perdidas
select * from PARTIDA; # Estado de partida 5 pasa a 'GANADOR JUGADOR2'

# Partida 6 estado 'TURNO JUGADOR1', si mueve en C2 futuro 'GANADOR JUGADOR1'
call mostrar_tablero(6);
select * from JUGADOR;
select * from PARTIDA;
update TABLERO set columnaC='X' where idPartida = 6 and fila = 2;
```

	PC4_INF_PRAC_01	30/10/20	Pràctica 1r DAM M02. Bases de dades		
			Professorat	Iciaraznar.prof	Pàgina 2 de 4

CFGS DESENVOLUPAMENT D'APLICACIONS MULTIPLATAFORMA - LOGÍSTICA			
MÒDUL 02: Bases de dades			
NOM I COGNOMS:			Nota:
UF 3: DCL i Rutines.	CURS: 22 - 23	Grup DAMIT	


```
call mostrar_tablero(6);
select * from JUGADOR; # Jugador rojo incrementa ganadas y jugador azul perdidas
select * from PARTIDA; # Estado de partida 6 pasa a 'GANADOR JUGADOR1'
```

```
# Partida 7 estado 'TURNO JUGADOR1', si mueve en A3 futuro 'GANADOR JUGADOR1'
call mostrar_tablero(7);
select * from JUGADOR;
select * from PARTIDA;
update TABLERO set columnaA='X' where idPartida = 7 and fila = 3;
call mostrar_tablero(7);
select * from JUGADOR; # Jugador rojo incrementa ganadas y jugador azul perdidas
select * from PARTIDA; # Estado de partida 7 pasa a 'GANADOR JUGADOR1'
```

P3) (3+1extra puntos) 3enRaya PROCEDURE:

```
# Procedure hacer_movimiento(in idPartidaP int, in loginJugadorP varchar(50), in passP varchar(50), in filaP int, in columnaP char(1), out
movimiento varchar(200))
# (3+1extra puntos)
# Generar un procedimiento que reciba en que partida (identificado por idPartida) que jugador (identificado por loginJugador y pass) quiere
marcar que casilla (identificada por fila y columna),
# y tras las verificaciones necesarias modifique tanto la casilla especificada en la tabla TABLERO, como modifique el estado para pasar el
turno al siguiente jugador en la tabla PARTIDA.
# Debe describir en un parametro de salida la accion realizada o el problema encontrado.
#
# En primer lugar, debe verificar si el loginJugador especificado como parametro de entrada existe en la tabla JUGADOR. Si no existe
loginJugador, especificar en el parametro de salida y finalizar.
# A continuacion, debe verificar si el pass del loginJugador, especificados como parametros de entrada, corresponden en la tabla
JUGADOR. Si no corresponden loginJugador y pass, especificar en el parametro de salida y finalizar.
# El paso siguiente es verificar si el idPartida especificado como parametro de entrada existe en la tabla PARTIDA. Si no existe la partida,
especificar en el parametro de salida y finalizar.
# A continuacion, debe verificar si el loginJugador especificado como parametro de entrada participa en la tabla PARTIDA especificada
como parametro de entrada. Si el jugador no participa, especificar en el parametro de salida y finalizar.
# El siguiente paso es verificar si el estado de la partida especificada como parametro de entrada en la tabla PARTIDA, identifica un turno
de jugador (contiene la palabra 'TURNO' en la tabla PARTIDA) o ya se ha finalizado la partida y no es turno de ningun jugador. Si el estado
no identifica un turno de jugador, especificar en el parametro de salida y finalizar.
# A continuacion, debe verificar si el estado de la partida especificada como parametro de entrada en la tabla PARTIDA, corresponde a
turno para el jugador que se está identificando (con su loginJugador como parámetro de entrada). Si segun el estado no es el turno de este
jugador, especificar en el parametro de salida y finalizar.
#
# Despues de estas verificaciones previas, para poder hacer el movimiento, todavia falta verificar si la fila y columna especificadas como
parametros de entrada son correctas (filas de 1 a 3, columnas 'A', 'B' o 'C'). Si no son coordenadas correctas, especificar en el parametro de
salida y finalizar.
# Tambien se debe verificar que la casilla del TABLERO (especificada por idPartida, fila y columna como parametros de entrada), este vacia
(con un espacio ' '). Si la casilla no esta vacia, especificar en el parametro de salida y finalizar.
#
# Por fin, se debe modificar la casilla del TABLERO (especificada por idPartida, fila y columna como parametros de entrada), con el simbolo
de 'X' si segun el estado es el turno de jugador1, o con el simbolo de 'O' si es el turno de jugador2.
# Y tambien, se debe modificar en la tabla PARTIDA el estado para pasar el turno al siguiente jugador (valorar el orden de estos dos ultimos
pasos).
# En el parametro de salida, si se ha podido realizar el movimiento, se especifica con el login de jugador.
#
# TEST hacer_movimiento CORRECTO
call hacer_movimiento(1, 'rojo', 'red', 2, 'C', @movimiento);
select @movimiento; # OK jugador rojo, movimiento realizado.
call mostrar_tablero(1);
```

```
call hacer_movimiento(1, 'azul', 'blue', 2, 'A', @movimiento);
select @movimiento; # OK jugador azul, movimiento realizado.
```

	PC4_INF_PRAC_01	30/10/20	Pràctica 1r DAM M02. Bases de dades		
			Professorat	Iciaraznar.prof	Pàgina 3 de 4


CFGS DESENVOLUPAMENT D'APLICACIONS MULTIPLATAFORMA - LOGÍSTICA			
MÒDUL 02: Bases de dades			
NOM I COGNOMS:			Nota:
UF 3: DCL i Rutines.	CURS: 22 - 23	Grup DAMIT	

call mostrar_tablero(1);

```

# TEST hacer_movimiento ERRORES DIVERSOS
call hacer_movimiento(1, 'rojos', 'red', 1, 'A', @movimiento);
select @movimiento; # ERROR: Jugador rojos NO existe en tabla JUGADOR.
call hacer_movimiento(1, 'rojo', 'reds', 1, 'A', @movimiento);
select @movimiento; # ERROR: Pass de jugador rojo NO corresponde en tabla JUGADOR.
call hacer_movimiento(10, 'rojo', 'red', 1, 'A', @movimiento);
select @movimiento; # ERROR: Partida 10 NO existe en tabla PARTIDA.
call hacer_movimiento(1, 'verde', 'green', 1, 'A', @movimiento);
select @movimiento; # ERROR: Jugador verde NO participa en partida 1 segun tabla PARTIDA.
call hacer_movimiento(2, 'verde', 'green', 1, 'A', @movimiento);
select @movimiento; # ERROR: NO es ESTADO de turno de NINGUN jugador en partida 2 segun tabla PARTIDA.
call hacer_movimiento(1, 'azul', 'blue', 1, 'A', @movimiento);
select @movimiento; # ERROR: NO es ESTADO de turno de jugador azul en partida 1 segun tabla PARTIDA.
call hacer_movimiento(1, 'rojo', 'red', 2, 'A', @movimiento);
select @movimiento; # ERROR: Casilla 2A NO vacia.
call hacer_movimiento(1, 'rojo', 'red', 1, 'D', @movimiento);
select @movimiento; # ERROR: Columna D NO valida.
call hacer_movimiento(1, 'rojo', 'red', 4, 'A', @movimiento);
select @movimiento; # ERROR: Fila 4 NO valida.

```

	PC4_INF_PRAC_01	30/10/20	Pràctica 1r DAM M02. Bases de dades		
			Professorat	Iciaraznar.prof	Pàgina 4 de 4