

FUNCIONES, PROCEDIMIENTOS, EVENTOS Y DISPARADORES EN LA BASE DE DATOS

Raúl Arribas Redondo

BBDD

1 Contenido

1.1	Crea una única rutina que sume todos los tantos marcados por los jugadores de cada clase, muestre el resultado con la suma y el nombre del grupo ordenado de mayor a menor y finalmente sume dos puntos al primer grupo y uno al segundo. Créala para que se utilicen los permisos del usuario ' limitado' @' localhost' para ejecutarla. ¿Qué permisos necesitará tener asociados ese usuario?.....	2
1.2	Crea un evento que llame cada semana al procedimiento anterior.....	7
1.3	Apartado 3:	9

- 1.1 Crea una **única** rutina que sume todos los tantos marcados por los jugadores de cada clase, muestre el resultado con la suma y el nombre del grupo ordenado de mayor a menor y finalmente sume dos puntos al primer grupo y uno al segundo. Créala para que se utilicen los permisos del usuario 'limitado'@'localhost' para ejecutarla. ¿Qué permisos necesitará tener asociados ese usuario?.

Con este condigo vamos a crear la base de datos baloncesto, sumará los tantos marcados por los jugadores de cada clase, muestra el resultado con la suma y ordena de mayor a menor puntuación, al primer grupo le añadirá dos puntos mas y al segundo que mas haya marcado, 1 punto.

Al final de script creamos el usuario limitado, y le otorgamos permisos para que pueda ejecutar la call de baloncesto que me hemos creado.

-- Usar la base de datos Baloncesto

```
drop database if exists baloncesto;
```

```
create database baloncesto;
```

```
USE Baloncesto;
```

-- Crear la tabla Clases

```
CREATE TABLE Clases (
```

```
    codigo CHAR(3) PRIMARY KEY,
```

```
    grupo VARCHAR(10) NOT NULL,
```

```
    puntuacion INT NOT NULL,
```

```
    capitan INT
```

```
) ENGINE=InnoDB;
```

-- Crear la tabla Jugadores

```
CREATE TABLE Jugadores (
```

```
    codalumno INT PRIMARY KEY,
```

```
    nombre VARCHAR(20) NOT NULL,
```

```
    apellido VARCHAR(20) NOT NULL,
```

```
    clase CHAR(3) NOT NULL,
```

```
    tantos_marcados INT NOT NULL,
```

```
    FOREIGN KEY (clase) REFERENCES Clases(codigo)
```

```
) ENGINE=InnoDB;
```

```
-- Insertar datos iniciales en la tabla Clases
```

```
INSERT INTO Clases (codigo, grupo, puntuacion, capitan) VALUES
```

```
('E1A', '1 ESO A', 6, 101),
```

```
('E1B', '1 ESO B', 2, 102),
```

```
('E2A', '2 ESO A', 0, 103),
```

```
('E2B', '2 ESO B', 4, 104);
```

```
-- Insertar datos iniciales en la tabla Jugadores
```

```
INSERT INTO Jugadores (codalumno, nombre, apellido, clase, tantos_marcados) VALUES
```

```
(101, 'Alvaro', 'Gomez', 'E1A', 50),
```

```
(102, 'Lucia', 'Martinez', 'E1B', 30),
```

```
(103, 'Mario', 'Lopez', 'E2A', 20),
```

```
(104, 'Sofia', 'Hernandez', 'E2B', 40),
```

```
(105, 'Elena', 'Fernandez', 'E1A', 30),
```

```
(106, 'Carlos', 'Perez', 'E1B', 20),
```

```
(107, 'Laura', 'Garcia', 'E2A', 10),
```

```
(108, 'Pedro', 'Rodriguez', 'E2B', 30);
```

```
-- Crear el procedimiento ActualizarPuntuaciones
```

```
DELIMITER $$
```

```
CREATE PROCEDURE ActualizarPuntuaciones()
```

```
BEGIN
```

```
-- Crear una tabla temporal con la suma de los tantos marcados por clase
```

```
CREATE TEMPORARY TABLE temp_clase_tantos AS
```

```
SELECT clase, SUM(tantos_marcados) AS total_tantos
```

```
FROM Jugadores
```

```
GROUP BY clase;
```

```
-- Identificar el primer y segundo grupo con más tantos marcados

SET @primer_grupo := (SELECT clase FROM temp_clase_tantos ORDER BY total_tantos DESC
LIMIT 1);

SET @segundo_grupo := (SELECT clase FROM temp_clase_tantos ORDER BY total_tantos
DESC LIMIT 1 OFFSET 1);


-- Actualizar puntuaciones en la tabla Clases

UPDATE Clases

SET puntuacion = puntuacion +

CASE

    WHEN codigo = @primer_grupo THEN 2

    WHEN codigo = @segundo_grupo THEN 1

    ELSE 0

END;


-- Eliminar la tabla temporal

DROP TEMPORARY TABLE temp_clase_tantos;

END$$


DELIMITER ;


-- Inicializar las puntuaciones en Clases (opcional)

UPDATE Clases

SET puntuacion = CASE

    WHEN codigo = 'E1A' THEN 6

    WHEN codigo = 'E1B' THEN 2

    WHEN codigo = 'E2A' THEN 0

    WHEN codigo = 'E2B' THEN 4

END;


-- Llamar al procedimiento para actualizar las puntuaciones
```

Funciones, procedimientos, eventos y disparadores en la base de datos

CALL ActualizarPuntuaciones();

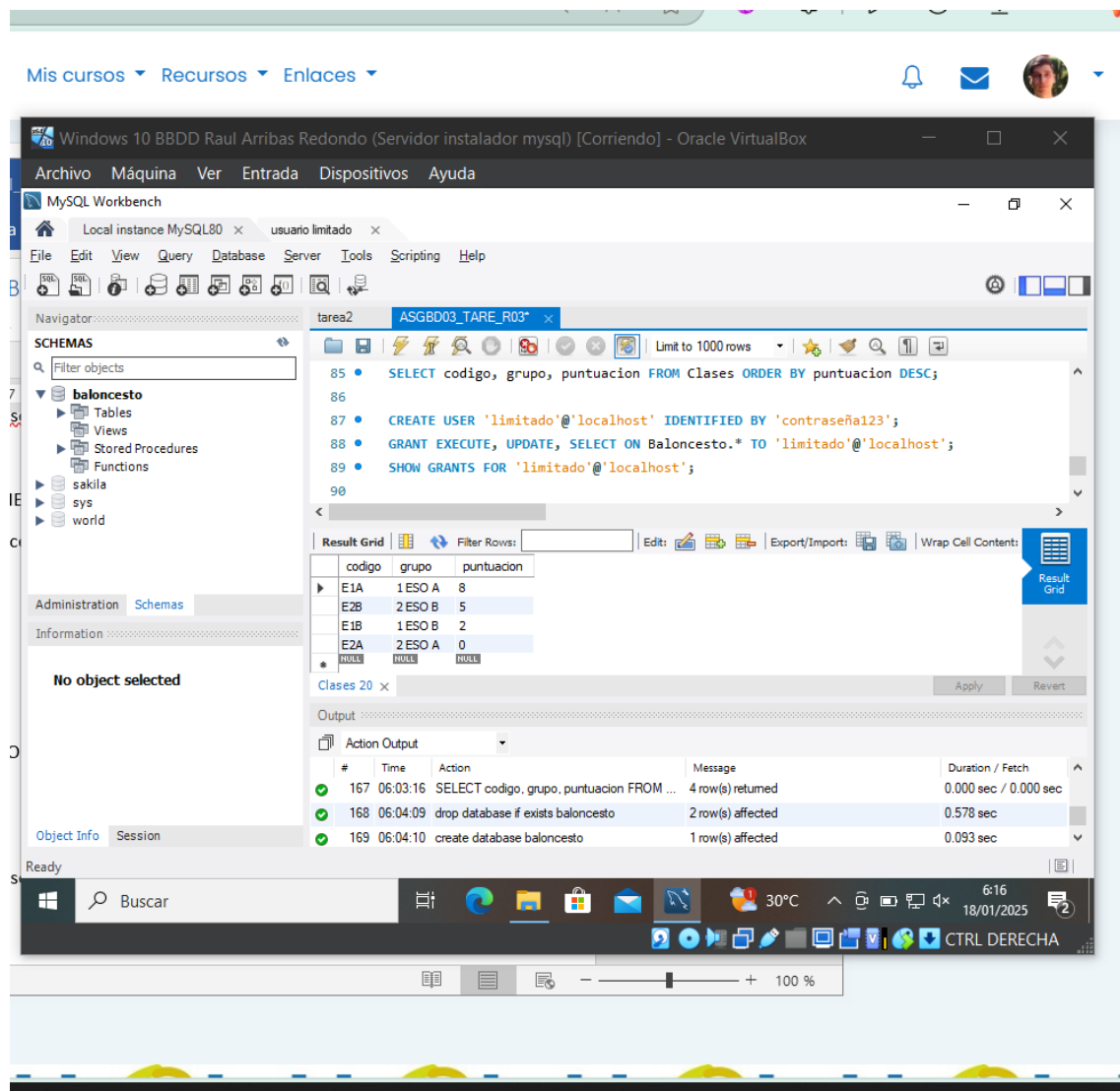
-- Consultar las puntuaciones actualizadas

SELECT codigo, grupo, puntuacion FROM Clases ORDER BY puntuacion DESC;

CREATE USER 'limitado'@'localhost' IDENTIFIED BY 'contraseña123';

GRANT EXECUTE, UPDATE, SELECT ON Baloncesto.* TO 'limitado'@'localhost';

SHOW GRANTS FOR 'limitado'@'localhost';



Funciones, procedimientos, eventos y disparadores en la base de datos

COMPROBACIÓN CON EL USUARIO LIMITADO:

use baloncesto;

CALL ActualizarPuntuaciones();

SELECT codigo, grupo, puntuacion FROM Clases ORDER BY puntuacion DESC;

The screenshot shows the MySQL Workbench interface. The 'Query 1' tab contains the following SQL script:

```
1 • use baloncesto;
2 • CALL ActualizarPuntuaciones();
3 • SELECT codigo, grupo, puntuacion FROM Clases ORDER BY puntuacion DESC;
```

The 'Result Grid' shows the output of the SELECT statement:

codigo	grupo	puntuacion
E1A	1 ESO A	8
E2B	2 ESO B	5
E1B	1 ESO B	2
E2A	2 ESO A	0
NULL	NULL	NULL

The 'Output' pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
3	05:52:08	CALL ActualizarPuntuaciones()	Error Code: 1175. You are using safe update...	0.016 sec
4	05:53:23	use baloncesto	0 row(s) affected	0.000 sec
5	05:53:23	CALL ActualizarPuntuaciones()	Error Code: 1050. Table 'tempo_clase tantos'...	0.000 sec

1.2 Crea un evento que llame cada semana al procedimiento anterior.

Creemos el evento para que cada semana llame al procedimiento ActualizarPuntuaciones

```
DELIMITER //
```

```
-- Crear el evento
```

```
CREATE EVENT ActualizarPuntuacionesSemanal
```

```
ON SCHEDULE EVERY 1 WEEK
```

```
STARTS CURRENT_TIMESTAMP
```

```
DO
```

```
BEGIN
```

```
    CALL ActualizarPuntuaciones();
```

```
END//
```

```
DELIMITER ;
```

```
SELECT EVENT_NAME, INTERVAL_VALUE, INTERVAL_FIELD, STARTS, STATUS
```

```
FROM information_schema.EVENTS
```

```
WHERE EVENT_NAME = 'ActualizarPuntuacionesSemanal';
```

Funciones, procedimientos, eventos y disparadores en la base de datos

Nos debería de salir esto para comprobar que funciona correctamente:

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'ActualizarPuntuaciones' under the 'baloncesto' database. The main editor shows a SQL script for a stored procedure and an event. The 'Result Grid' at the bottom displays the details of the 'ActualizarPuntuacionesSemanal' event.

```
103 BEGIN
104     CALL ActualizarPuntuaciones();
105 END//
106
107 DELIMITER ;
108
109 • SELECT EVENT_NAME, INTERVAL_VALUE, INTERVAL_FIELD, STARTS, STATUS
110 FROM information_schema.EVENTS
111 WHERE EVENT_NAME = 'ActualizarPuntuacionesSemanal';
```

EVENT_NAME	INTERVAL_VALUE	INTERVAL_FIELD	STARTS	STATUS
ActualizarPuntuacionesSemanal	1	WEEK	2025-01-18 16:09:40	ENABLED

Output:

#	Time	Action	Message	Duration / Fetch
✓ 100	16:09:40	CALL ActualizarPuntuaciones()	0 row(s) affected	0.000 sec

1.3 Apartado 3:

Creemos la tabla HCO_CAPITANES y agregamos dos columnas a la tabla clases:

```
drop table if exists HCO_CAPITANES;
```

```
CREATE TABLE HCO_CAPITANES (  
    id INT AUTO_INCREMENT PRIMARY KEY,  
    nomantc VARCHAR(50), -- Nombre del capitán anterior  
    apeantc VARCHAR(50), -- Apellido del capitán anterior  
    nomactc VARCHAR(50), -- Nombre del capitán actual  
    apeactc VARCHAR(50), -- Apellido del capitán actual  
    clase VARCHAR(10), -- Código de la clase donde cambia el capitán  
    fecambio DATETIME -- Fecha del cambio de capitán  
);
```

```
ALTER TABLE Clases
```

```
ADD COLUMN capitan_nombre VARCHAR(50),
```

```
ADD COLUMN capitan_apellido VARCHAR(50);
```

The screenshot shows a MySQL Workbench window titled "ASGBD03_TARE_R03*" with a tab for "es2". The interface includes a menu bar (Tools, Scripting, Help), a toolbar, and a main editor area. The editor contains the following SQL code:

```
13 • SELECT EVENT_NAME, EVENT_DEFINITION, STATUS, STARTS, INTERVAL_VALUE, INTERVAL_FIELD, LAST_EXECUTED
14 FROM information_schema.EVENTS
15 WHERE EVENT_NAME = 'ActualizarPuntuacionesSemanal';
16
17 • CREATE TABLE HCO_CAPITANES (
18   id INT AUTO_INCREMENT PRIMARY KEY,
19   nomantc VARCHAR(50), -- Nombre del capitán anterior
20   apeantc VARCHAR(50), -- Apellido del capitán anterior
21   nomactc VARCHAR(50), -- Nombre del capitán actual
22   apeactc VARCHAR(50), -- Apellido del capitán actual
23   clase VARCHAR(10), -- Código de la clase donde cambia el capitán
24   fecambio DATETIME -- Fecha del cambio de capitán
25 );
26
27
```

Below the editor is the "Output" pane, which shows the "Action Output" for the executed queries:

#	Time	Action	Message	Duration / Fetch
111	16:15:54	SELECT EVENT_NAME, NEXT_EXECUTION, STATUS FROM information_schema.EVENTS WHERE EVENT_NAME = 'ActualizarPuntuacionesSemanal';	Error Code: 1054. Unknown table 'ActualizarPuntuacionesSemanal'	0.000 sec
112	16:16:04	SHOW VARIABLES LIKE 'event_scheduler';	1 row(s) returned	0.000 sec / 0.000 sec
113	16:17:03	SELECT EVENT_NAME, EVENT_DEFINITION, STATUS, STARTS, INTERVAL_VALUE, INTERVAL_FIELD, LAST_EXECUTED FROM information_schema.EVENTS WHERE EVENT_NAME = 'ActualizarPuntuacionesSemanal';	1 row(s) returned	0.000 sec / 0.000 sec
114	16:20:00	CREATE TABLE HCO_CAPITANES (id INT AUTO_INCREMENT PRIMARY KEY, nomantc VARCHAR(50), -- N...)	0 row(s) affected	0.266 sec

The Windows taskbar at the bottom shows the system clock as 16:20 on 18/01/2025, along with various application icons and a "CTRL DERECHA" notification.

Creemos el Trigger, el cual se activará después de que se actualice el capitán de una clase en la tabla Clases:

```
DELIMITER //
```

```
CREATE TRIGGER TriggerCambioCapitan
```

```
AFTER UPDATE ON Clases
```

```
FOR EACH ROW
```

```
BEGIN
```

```
-- Comprobar si el capitán ha cambiado
```

```
IF OLD.capitan_nombre != NEW.capitan_nombre OR OLD.capitan_apellido !=  
NEW.capitan_apellido THEN
```

```
-- Insertar un registro en HCO_CAPITANES con los datos anteriores y actuales
```

```
INSERT INTO HCO_CAPITANES (nomantc, apeantc, nomactc, apeactc, clase, fecambio)
```

```
VALUES (OLD.capitan_nombre, OLD.capitan_apellido, NEW.capitan_nombre,  
NEW.capitan_apellido, NEW.codigo, NOW());
```

```
END IF;
```

```
END//
```

```
DELIMITER ;
```

Comprobación de que funciona: *TENEMOS QUE CAMBIAR UN NOMBRE POR OTRO Y HACER DOS UPDATES, PARA QUE SALGA EL ANTERIOR Y EL NUEVO*

```
UPDATE Clases
```

```
SET capitan_nombre = 'Pedro', capitan_apellido = 'González'
```

```
WHERE codigo = 'E1A';
```

```
UPDATE Clases
```

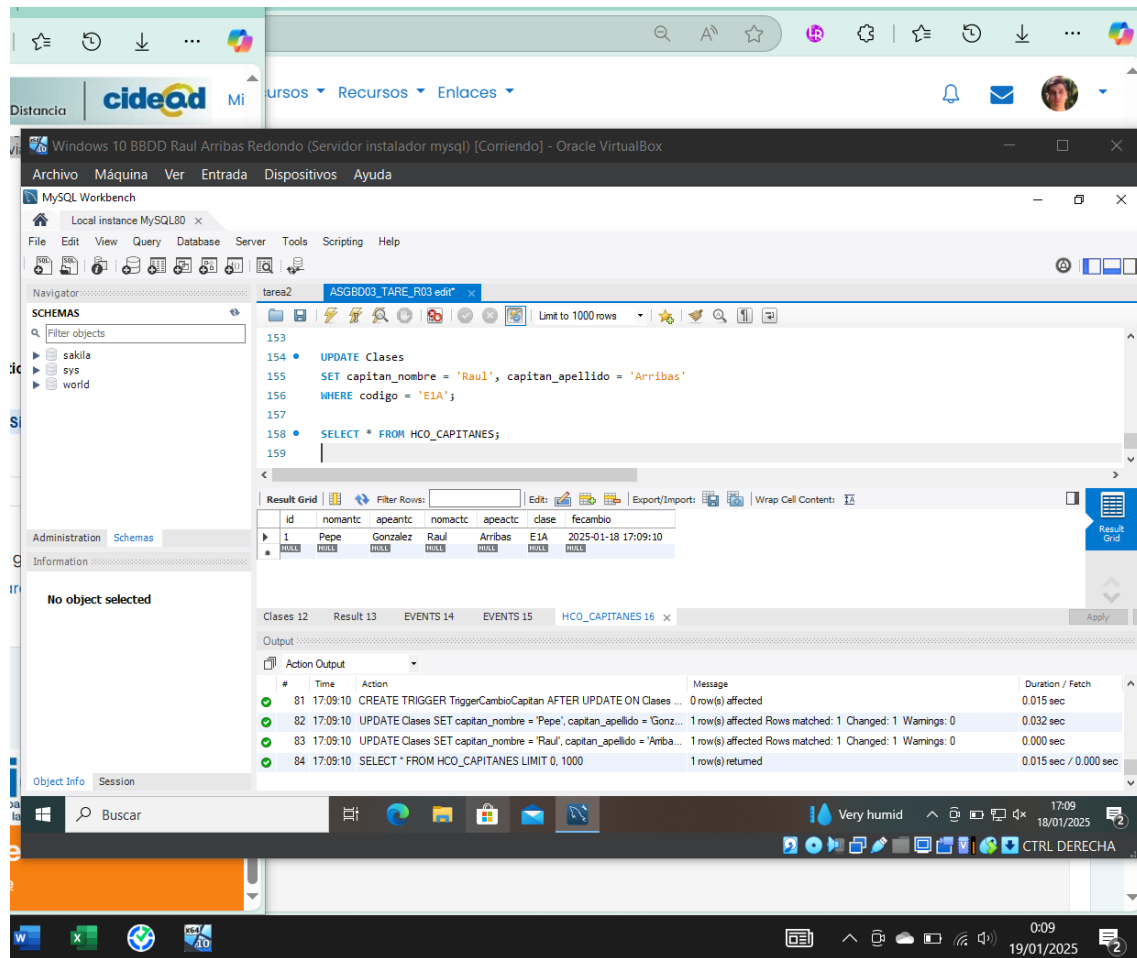
```
SET capitan_nombre = Raul, capitan_apellido = 'Arribas'
```

```
WHERE codigo = 'E1A';
```

```
SELECT * FROM HCO_CAPITANES;
```

Funciones, procedimientos, eventos y disparadores en la base de datos

Esta es la comprobación con una captura de pantalla, que al cambiar de capitán en una clase se actualiza:



SCRIPT COMPLETO:

-- Usar la base de datos Baloncesto

drop database if exists baloncesto;

create database baloncesto;

USE Baloncesto;

-- Crear la tabla Clases

CREATE TABLE Clases (

codigo CHAR(3) PRIMARY KEY,

grupo VARCHAR(10) NOT NULL,

puntuacion INT NOT NULL,

capitan INT

) ENGINE=InnoDB;

-- Crear la tabla Jugadores

CREATE TABLE Jugadores (

codalumno INT PRIMARY KEY,

nombre VARCHAR(20) NOT NULL,

apellido VARCHAR(20) NOT NULL,

clase CHAR(3) NOT NULL,

tantos_marcados INT NOT NULL,

FOREIGN KEY (clase) REFERENCES Clases(codigo)

) ENGINE=InnoDB;

-- Insertar datos iniciales en la tabla Clases

INSERT INTO Clases (codigo, grupo, puntuacion, capitan) VALUES

('E1A', '1 ESO A', 6, 101),

('E1B', '1 ESO B', 2, 102),

('E2A', '2 ESO A', 0, 103),

('E2B', '2 ESO B', 4, 104);

-- Insertar datos iniciales en la tabla Jugadores

```
INSERT INTO Jugadores (codalumno, nombre, apellido, clase, tantos_marcados) VALUES
(101, 'Alvaro', 'Gomez', 'E1A', 50),
(102, 'Lucia', 'Martinez', 'E1B', 30),
(103, 'Mario', 'Lopez', 'E2A', 20),
(104, 'Sofia', 'Hernandez', 'E2B', 40),
(105, 'Elena', 'Fernandez', 'E1A', 30),
(106, 'Carlos', 'Perez', 'E1B', 20),
(107, 'Laura', 'Garcia', 'E2A', 10),
(108, 'Pedro', 'Rodriguez', 'E2B', 30);
```

-- Crear el procedimiento ActualizarPuntuaciones

DELIMITER \$\$

CREATE PROCEDURE ActualizarPuntuaciones()

BEGIN

-- Crear una tabla temporal con la suma de los tantos marcados por clase

CREATE TEMPORARY TABLE temp_clase_tantos AS

SELECT clase, SUM(tantos_marcados) AS total_tantos

FROM Jugadores

GROUP BY clase;

-- Identificar el primer y segundo grupo con más tantos marcados

SET @primer_grupo := (SELECT clase FROM temp_clase_tantos ORDER BY total_tantos DESC
LIMIT 1);

SET @segundo_grupo := (SELECT clase FROM temp_clase_tantos ORDER BY total_tantos
DESC LIMIT 1 OFFSET 1);

-- Actualizar puntuaciones en la tabla Clases

UPDATE Clases

SET puntuacion = puntuacion +

```
CASE
```

```
    WHEN codigo = @primer_grupo THEN 2
```

```
    WHEN codigo = @segundo_grupo THEN 1
```

```
    ELSE 0
```

```
END;
```

```
-- Eliminar la tabla temporal
```

```
DROP TEMPORARY TABLE temp_clase_tantos;
```

```
END$$
```

```
DELIMITER ;
```

```
-- Inicializar las puntuaciones en Clases (opcional)
```

```
UPDATE Clases
```

```
SET puntuacion = CASE
```

```
    WHEN codigo = 'E1A' THEN 6
```

```
    WHEN codigo = 'E1B' THEN 2
```

```
    WHEN codigo = 'E2A' THEN 0
```

```
    WHEN codigo = 'E2B' THEN 4
```

```
END;
```

```
-- Llamar al procedimiento para actualizar las puntuaciones
```

```
SET SQL_SAFE_UPDATES = 0;
```

```
CALL ActualizarPuntuaciones();
```

```
-- Consultar las puntuaciones actualizadas
```

```
SELECT codigo, grupo, puntuacion FROM Clases ORDER BY puntuacion DESC;
```

```
DROP USER IF EXISTS 'limitado'@'localhost';
```

```
CREATE USER 'limitado'@'localhost' IDENTIFIED BY 'contraseña123';
```

```
GRANT EXECUTE, UPDATE, SELECT ON Baloncesto.* TO 'limitado'@'localhost';
```

```
SHOW GRANTS FOR 'limitado'@'localhost';
```

```
DELIMITER //
```

```
-- Eliminar el evento si ya existe
```

```
-- Crear el evento
```

```
CREATE EVENT ActualizarPuntuacionesSemanal
```

```
ON SCHEDULE EVERY 1 WEEK
```

```
STARTS CURRENT_TIMESTAMP
```

```
DO
```

```
BEGIN
```

```
    CALL ActualizarPuntuaciones();
```

```
END//
```

```
DELIMITER ;
```

```
SELECT EVENT_NAME, INTERVAL_VALUE, INTERVAL_FIELD, STARTS, STATUS
```

```
FROM information_schema.EVENTS
```

```
WHERE EVENT_NAME = 'ActualizarPuntuacionesSemanal';
```

```
SELECT EVENT_NAME, EVENT_DEFINITION, STATUS, STARTS, INTERVAL_VALUE,
```

```
INTERVAL_FIELD, LAST_EXECUTED
```

```
FROM information_schema.EVENTS
```

```
WHERE EVENT_NAME = 'ActualizarPuntuacionesSemanal';
```

```
drop table if exists HCO_CAPITANES;
```

```
CREATE TABLE HCO_CAPITANES (
```

```
    id INT AUTO_INCREMENT PRIMARY KEY,
```

```
    nomantc VARCHAR(50), -- Nombre del capitán anterior
```

```
apeantc VARCHAR(50), -- Apellido del capitán anterior
nomactc VARCHAR(50), -- Nombre del capitán actual
apeactc VARCHAR(50), -- Apellido del capitán actual
clase VARCHAR(10), -- Código de la clase donde cambia el capitán
fecambio DATETIME -- Fecha del cambio de capitán
);
```

ALTER TABLE Clases

```
ADD COLUMN capitan_nombre VARCHAR(50),
ADD COLUMN capitan_apellido VARCHAR(50);
```

```
SET SQL_SAFE_UPDATES = 0;
```

```
DROP TRIGGER IF EXISTS TriggerCambioCapitan;
```

```
DELIMITER //
```

```
CREATE TRIGGER TriggerCambioCapitan
```

```
AFTER UPDATE ON Clases
```

```
FOR EACH ROW
```

```
BEGIN
```

```
-- Comprobar si el capitán ha cambiado
```

```
IF OLD.capitan_nombre != NEW.capitan_nombre OR OLD.capitan_apellido !=
NEW.capitan_apellido THEN
```

```
-- Insertar un registro en HCO_CAPITANES con los datos anteriores y actuales
```

```
INSERT INTO HCO_CAPITANES (nomantc, apeantc, nomactc, apeactc, clase, fecambio)
```

```
VALUES (OLD.capitan_nombre, OLD.capitan_apellido, NEW.capitan_nombre,
NEW.capitan_apellido, NEW.codigo, NOW());
```

```
END IF;
```

```
END//
```

```
DELIMITER ;
```

Funciones, procedimientos, eventos y disparadores en la base de datos

UPDATE Clases

SET capitan_nombre = 'Pepe', capitan_apellido = 'Gonzalez'

WHERE codigo = 'E1A';

UPDATE Clases

SET capitan_nombre = 'Raul', capitan_apellido = 'Arribas'

WHERE codigo = 'E1A';

SELECT * FROM HCO_CAPITANES;