

1- Crea una única rutina que sume todos los tantos marcados por los jugadores de cada clase, muestre el resultado de 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.

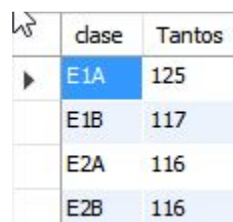
NOTA: *antes de nada debemos actualizar algunos registros, puesto que el resultado de la suma sale empate. Lo hacemos con la sentencia update sobre la tabla jugadores:*

UPDATE jugadores SET tantos_marcados = 25 WHERE codalumno='E1A016';

UPDATE jugadores SET tantos_marcados = 17 WHERE codalumno='E1B603';

Tantos marcados ordenados de mayor a menor:

*SELECT clase, SUM(tantos_marcados) AS Tantos
FROM jugadores
GROUP BY clase
ORDER BY SUM(tantos_marcados) DESC;*



clase	Tantos
E1A	125
E1B	117
E2A	116
E2B	116

Crearemos una nueva tabla llamada BORRAME que tal y como su nombre indica, la borraremos al finalizar el procedimiento y en la que además capturaremos los datos de la anterior consulta para trabajar con ella:

*CREATE TABLE BORRAME AS
SELECT clase, SUM(tantos_marcados) AS Tantos
FROM jugadores GROUP BY clase
ORDER BY SUM(tantos_marcados) DESC;*

Una vez tenemos la nueva tabla empezaremos a trabajar con ella. Cribamos los resultados de la nueva tabla “BORRAME” a “1” para obtener unicamente la clase con más tantos e introducimos esa información en la variable G1:

SELECT clase FROM BORRAME LIMIT 1 INTO G1;

Ahora realizamos la misma operación pero la criba la haremos con el segundo resultado más alto “1,1” e introducimos la información en la variable G2:

SELECT clase FROM BORRAME LIMIT 1, 1 INTO G2;

Ahora procedemos a generar la rutina incluyendo las anteriores sentencias:

DELIMITER //

DROP PROCEDURE IF EXISTS ejercicio1 //

CREATE DEFINER = 'limitado'@'localhost' PROCEDURE ejercicio1()

BEGIN

Declaramos las variables

DECLARE G1 VARCHAR(3);

DECLARE G2 VARCHAR(3);

Creamos la tabla BORRAME e introducimos los campos de grupo, clase y la suma de los tantos marcados

CREATE TABLE BORRAME AS

SELECT grupo, clase, SUM(tantos_marcados) AS Tantos

FROM jugadores, clases

WHERE clases.codigo = jugadores.clase

GROUP BY clase ORDER BY SUM(tantos_marcados) DESC;

Introducimos el campo clase de la tabla BORRAME en la variable G1 y G2 y hacemos los correspondientes UPDATE con la información de las variables.

SELECT clase FROM BORRAME LIMIT 1 INTO G1;

SET SQL_SAFE_UPDATES=0;

Actualizamos la tabla BORRAME con los datos introducidos en G1

UPDATE BORRAME SET Tantos = Tantos + 2 WHERE clase = G1;

SELECT clase FROM BORRAME LIMIT 1, 1 INTO G2;

UPDATE BORRAME SET Tantos = Tantos + 1 WHERE clase = G2;

SET SQL_SAFE_UPDATES=1;

Seleccionamos los campos grupo y tantos de la tabla BORRAME y la borramos

SELECT grupo, tantos FROM BORRAME;

DROP TABLE IF EXISTS BORRAME;

END

//

DELIMITER ;

Ahora llamamos al procedimiento y comprobamos si funciona correctamente:

call ejercicio1();

	grupo	tantos
▶	1 ESO A	127
	1 ESO B	118
	2 ESO A	116
	2 ESO B	116

Podemos observar, que comparando la captura de la primera sentencia con el resultado del procedimiento, se le ha sumado dos tantos al primer grupo y un tanto al segundo.

Hemos utilizado una tabla temporal denominada “BORRAME” y hemos insertado en ella los datos **grupo**, **clase** y **SUM(tantos_marcados)** de la tabla jugadores. Una vez hecho hemos utilizado dos consultas para cribar esa información obteniendo el primer y el segundo grupo con más tantos e introduciendo esa información en las variables **G1** y **G2**. Después hemos actualizado los tantos con **UPDATE** tomando la información de **G1** y **G2** respectivamente sumándole “2 tantos” a **G1**(grupo con más tantos) y “1 tanto” a **G2**(segundo grupo con más tantos).

Además, hemos tenido que modificar la opción **SAFE UPDATE** puesto que está habilitada por defecto.

¿Qué permisos necesitará tener asociados este usuario?

- El primer paso es crear el usuario (lo hemos hecho previamente):

```
CREATE USER 'limitado'@'localhost';
```

- El siguiente paso es darle permisos al usuario para que pueda ejecutar SELECT, CREATE, DROP y UPDATE:

```
GRANT SELECT, UPDATE, CREATE, DROP ON Baloncesto.* TO  
'limitado'@'localhost' IDENTIFIED BY 'localhost';
```

2- Crea un evento que llame cada semana al procedimiento anterior

```
# Se ejecutará el día 12 de cada mes a las 9:00
CREATE EVENT ejercicio2
ON SCHEDULE
EVERY 1 WEEK STARTS '2017-02-12 9:00:00'
COMMENT 'Realizamos comprobación puntuación'
DO CALL ejercicio1();
```

Mostramos el evento:

```
SHOW EVENTS;
```

Db	Name	Definer	Time zone	Type	Execute at	Interval value	Interval field	Starts	Ends	Status	Originator	character_set_client	collation_connection	Database Collation
baloncesto	ejercicio2	root@localhost	SYSTEM	RECURRING	NULL	1	WEEK	2017-02-12 09:00:00	NULL	ENABLED	1	utf8	utf8_general_ci	utf8_general_ci

3- Crea la tabla HCO_CAPITANES con la estructura siguiente:


- **id:** entero, autoincremental, PK
- **nomantc:** Nombre del capitán anterior
- **apeantc:** Apellido del capitán anterior
- **nomactc:** Nombre del capitán actual
- **apeactc:** Apellido del capitán actual
- **clase:** Código de la clase en la que cambia el capitán
- **fecambio:** Fecha del cambio de capitán

Todos los campos son obligatorios.

Codifica un disparador que inserte un registro en HCO_CAPITANES tras cambiar el capitán de una clase.

El primer paso es crear la tabla en la base de datos “Baloncesto”:

```
create table HCO_CAPITANES(
    id int(11) primary key not null AUTO_INCREMENT,
    nomantc varchar(40) not null,
    apeantc varchar(40) not null,
    nomactc varchar(40) not null,
    apeactc varchar(40) not null,
    clase varchar(40) not null,
    fecambio datetime not null
) engine = innodb;
```

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Extra
<input type="checkbox"/> 1	id 	int(11)			No	Ninguna	AUTO_INCREMENT
<input type="checkbox"/> 2	nomantc	varchar(40)	utf8_general_ci		No	Ninguna	
<input type="checkbox"/> 3	apeatc	varchar(40)	utf8_general_ci		No	Ninguna	
<input type="checkbox"/> 4	nomactc	varchar(40)	utf8_general_ci		No	Ninguna	
<input type="checkbox"/> 5	apectc	varchar(40)	utf8_general_ci		No	Ninguna	
<input type="checkbox"/> 6	clase	varchar(40)	utf8_general_ci		No	Ninguna	
<input checked="" type="checkbox"/> 7	fecambio	datetime			No	Ninguna	

Ahora pasamos a crear el trigger:

```
DELIMITER //
```

```
DROP TRIGGER IF EXISTS ejercicio3 //
```

```
CREATE TRIGGER ejercicio3 BEFORE UPDATE ON clases FOR EACH ROW
```

```
BEGIN
```

```
# Declaramos 4 variables donde introduciremos el nombre antiguo/nuevo y Apellido  
antiguo/nuevo
```

```
    DECLARE OLDnombre VARCHAR(25);
```

```
    DECLARE OLDapellido VARCHAR(25);
```

```
    DECLARE NEWnombre VARCHAR(25);
```

```
    DECLARE NEWapellido VARCHAR(25);
```

```
# Introducimos nombre y apellido de la tabla jugadores dentro de las variables.
```

```
    SELECT nombre, apellido INTO OLDnombre, OLDapellido FROM jugadores
```

```
    WHERE codalumno = OLD.capitan;
```

```
    SELECT nombre, apellido INTO NEWnombre, NEWapellido FROM jugadores
```

```
    WHERE codalumno = NEW.capitan;
```

```
# Insertamos en los campos de hco_capitanes los nuevos valores
```

```
    INSERT INTO HCO_CAPITANES
```

```
    (nomantc, apeantc, nomactc, apectc, clase, fecambio)
```

```
    VALUES
```

```
    (OLDnombre, OLDAPELLIDO, NEWnombre, NEWapellido, NEW.codigo, NOW());
```

```
END //
```

```
DELIMITER ;
```

Antes de cambiar un capitán de la clase para hacer la comprobación mostraremos los que lo son actualmente:

```
SELECT capitan, nombre, apellido FROM jugadores, clases
WHERE jugadores.codalumno = clases.capitan;
```

	capitan	nombre	apellido
▶	E1A777	ALEPH	ONSO
	E1B996	NAZARIUS	FLINT
	E2A655	MORGANA	PENDRAGON
	E2B676	PAUL	FONTOTHE

Una vez sabemos los nombres y apellidos de los capitanes de la clase, vamos a cambiarlo por otro alumno para comprobar que el TRIGGER hace su trabajo:

```
UPDATE clases SET capitan='E1A016' WHERE capitan='E1A777';
```

codalumno	nombre	apellido	tantos_marcados	puesto	clase
E1A016	ALBUS	DEKA	16	4	E1A
E1A603	ENRIQUE	ALFARERO	16	2	E1A
E1A666	OLGA	SCOTT	16	3	E1A
E1A689	JOHNNY	BERTO	16	5	E1A
E1A776	MELVIN	SQUIRRELS	20	1	E1A
E1A777	ALEPH	ONSO	16	3	E1A
E1A888	PAUVAR	ELA	16	1	E1A
E1B016	SEVERIUS	STUKA	16	4	E1B
E1B603	DRACO	MALFOY	16	1	E1B
E1B666	ALF	MELMAC	16	3	E1B
E1B689	LORDPARTH	MADER	16	1	E1B

...y comprobamos si se ha insertado los registros pertinentes en la tabla HCO CAPITANES:

```
SELECT * FROM hco capitanes;
```

[illegible]