Vamos a ver mediante un ejemplo de que los procedimientos, en muchos casos, pueden utilizarse también para introducir datos en tablas, modificarlos o borrarlos, a través de la introducción de parámetros que me indiquen qué elementos quiero añadir, modificar o borrar.

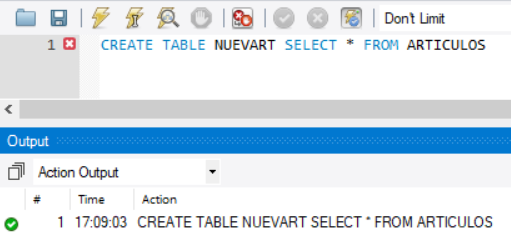
Por supuesto que, antes de realizar dicha operación, será necesario comprobar si el dato que quiero introducir, por ejemplo, ya existe o si el dato a modificar o borrar no existe.

Daos cuenta de que si yo intento introducir una fila en una tabla que ya existe en la misma, probablemente estemos intentando duplicar la clave y eso no nos lo va a permitir.

Se nos pueden dar 3 posibles casos:

1. **Que me manden crear un procedimiento para introducir datos en una tabla 🡪** Tendré que comprobar primero que el dato no existe en la tabla, para poder añadirlo.
2. **Que me manden crear un procedimiento para modificar datos en una tabla 🡪** Tendré que comprobar primero que el dato existe para poder modificarlo.
3. **Que me manden borrar datos en una tabla 🡪** Tendré que comprobar primero que el dato existe para poder borrarlo.

Veamos un ejemplo. Dentro del esquema supermercado, vamos a hacer una copia de la tabla artículos y la vamos a llamar NUEVART.

****

**EJEMPLO DE PROCEDIMIENTO PARA INSERTAR UN NUEVO ARTÍCULO EN LA TABLA ARTÍCULOS DEL ESQUEMA SUPERMERCADO**

Vamos a crear ahora un procedimiento que permita insertar un nuevo artículo en la tabla artículos del esquema supermercado. Los campos a insertar (código de artículo, nombre artículo, departamento del artículo y precio) serán los parámetros del procedimiento. Antes de insertar, tendremos que ver si dicho artículo existe ya. Daos cuenta de que podemos insertar dentro de un mismo departamento dos artículos que pueden llamarse igual, o tener un mismo precio, pero lo que nunca va a poder coincidir es el código de artículo. Este campo es la clave, y es lo que vamos a tener que comprobar antes de introducir el artículo. Por supuesto que, si yo no hago la comprobación, me va a dar un error de clave duplicada, pero puede interesarme que en lugar de mostrar dicho error abajo, me aparezca un mensaje por pantalla, indicándome que ese artículo ya existe.

Vamos a crear el procedimiento:

**CREATE DEFINER=`root`@`localhost` PROCEDURE `INTRODUCEARTIC`(C INT, N VARCHAR(40),D INT, P DECIMAL(18,2))**

**BEGIN**

**DECLARE e INT;**

**SELECT CODART into e FROM NUEVART**

**where C=CODART;**

**IF e IS NULL THEN**

**INSERT INTO NUEVART VALUES (C,N,D,P);**

**ELSE**

**select "CÓDIGO DE ARTÍCULO DUPLICADO";**

**END IF;**

**END**

El bloque que está sombreado en amarillo se podría poner también de la siguiente manera:

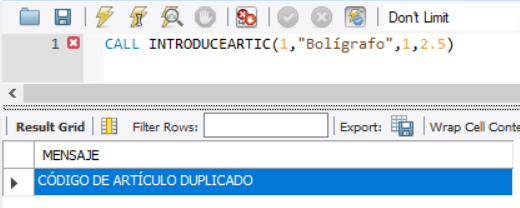
**DECLARE e INT;**

**SET e=(SELECT CODART FROM NUEVART where C=CODART);**

**Podéis usar la forma que más os guste o que más sencilla os resulte.**

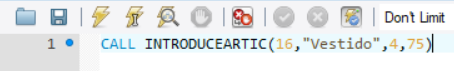
Lo que estamos haciendo es buscar en la tabla CODART el parámetro C que se corresponde con el código del artículo a insertar. Si lo encontramos se guarda en e. Si no lo encuentra e toma valor nulo. Es en este último caso, cuando mando insertar el artículo en la tabla. Si lo encuentra, como el código de artículo está repetido, mostrará un mensaje de error.

Al ejecutar, obtendremos lo siguiente:



Como he ejecutado poniendo el código de artículo 1, que ya existía, me muestra el mensaje de error.

Vamos a ver ahora qué ocurre si introduzco un código de artículo que no existe.





**EJEMPLO DE PROCEDIMIENTO PARA ELIMINAR UN ARTÍCULO EN LA TABLA ARTÍCULOS DEL ESQUEMA SUPERMERCADO**

Vamos ahora a crear un procedimiento que elimine un artículo de la tabla NUEVART. Introduciremos el código de artículo a eliminar como parámetro del procedimiento. Queremos que si el código de artículo introducido no existe, me aparezca en pantalla el siguiente mensaje de error: “Ese código de artículo no existe”.

En este caso, tendremos que borrar cuando encontremos el código, es decir, cuando la variable donde volcamos dicho código no sea nula.

El procedimiento sería el siguiente:

**CREATE DEFINER=`root`@`localhost` PROCEDURE `BORRART`(X int)**

**BEGIN**

**DECLARE e int;**

**SELECT CODART into e FROM nuevart**

**where x=CODART;**

**IF e IS NULL THEN**

**SELECT "Ese código de artículo no existe" as resultado;**

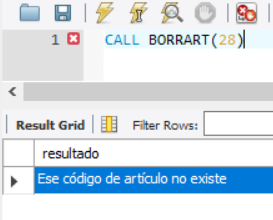
**else**

**DELETE FROM nuevart WHERE CODART=X;**

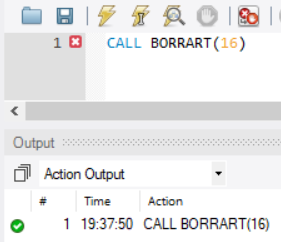
**end if;**

**END**

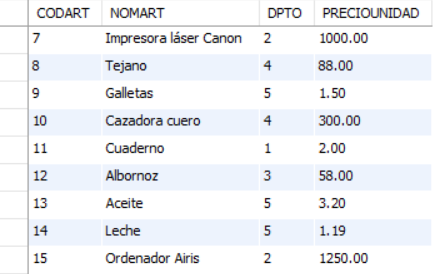
Si lo ejecuto con un código de artículo que no existe el resultado será el siguiente:



En el caso de meter un código de artículo a borrar que sí exista, el artículo se borrará como se observa a continuación.



Observamos que el artículo ya no aparece:



Debes tener en cuenta que **el parámetro tienes que buscarlo en la tabla en la que dicho elemento no se repita, es decir, funcione como clave.** Me explico. Un código de departamento puedes encontrarlo dentro de la tabla departamento y dentro de la tabla empleado. En la tabla departamento funciona como clave y sólo aparece una vez. Sin embargo, en la tabla empleado puede repetirse, puesto que puede haber más de un empleado en cada departamento. En ese caso, se intentaría volcar más de un valor en la variable.

Si yo tuviera el siguiente procedimiento:

CREATE DEFINER=`root`@`localhost` PROCEDURE `INSERDEPAR`(C INT, L VARCHAR(20), N VARCHAR(14))

BEGIN

DECLARE e INT;

SELECT DEPNUM into e FROM EMPLEADO

where C=DEPNUM;

IF e IS NULL THEN

SELECT "DEPARTAMENTO NO ENCONTRADO" AS MENSAJE;

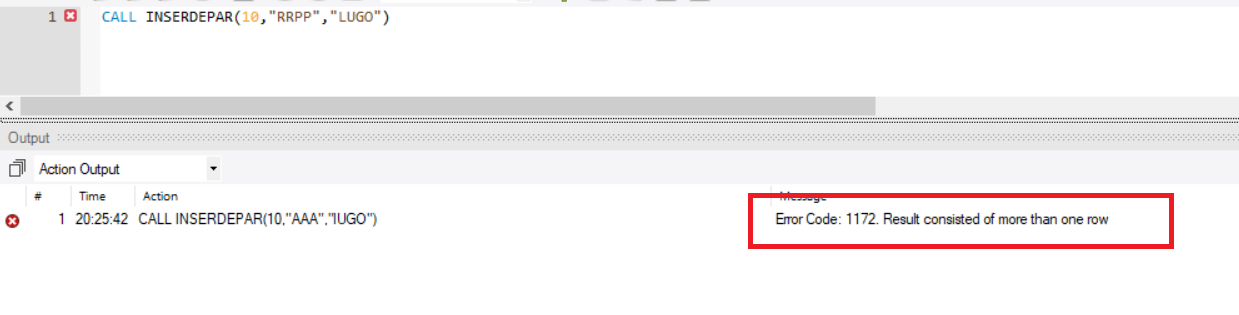
ELSE

select "DEPARTAMENTO DUPLICADO" as MENSAJE;

END IF;

END

Y lo ejecutara de la siguiente manera:



Vemos que obtenemos el error de que el resultado devuelva más de una fila. Ello se debe al bloque siguiente:

SELECT DEPNUM into e FROM EMPLEADO

where C=DEPNUM;

Vemos que la tabla que hemos puesto es la de EMPLEADO, y ahí el número de departamento puede aparecer más de una vez porque hay varios empleados que pertenecen al mismo.

Para que saliera bien, tendríamos que poner:

SELECT DEPNUM into e FROM DEPARTAMENTO

where C=DEPNUM;

puesto que la tabla DEPARTAMENTO es la que contiene el número de departamento de forma única, puesto que es la clave.