# BD UD6 Procedimientos

Introducción

* Un procedimiento almacenado es un recurso que incorporan todos los gestores relacionales y que permite asociar un nombre (el nombre del procedimiento) a un conjunto de instrucciones SQL.

Cada vez que se llama al procedimiento, se ejecutan todas las instrucciones.

* A diferencia de un método en programación orientada a objetos, podremos emplear instrucciones SQL que serán ejecutadas directamente en el gestor.
* Normalmente al desarrollar una aplicación, tendremos:
* Un frontal cliente (es decir, la pantalla/s que usa el cliente para obtener y realizar operaciones contra una base de datos). Este programa cliente estará instalado en cada uno los ordenadores de los usuarios (si es una aplicación de escritorio, como una realizada con NetBeans, VC#, VB,...)
* Un servidor donde está instalado el gestor y donde se encuentra la base de datos a la que los usuarios acceden (normalmente en un único ordenador).

Dentro del código fuente que conforma el programa del cliente es donde vamos a ejecutar las órdenes SQL de consulta o modificación de los datos (SELECT, UPDATE, DELETE o INSERT) así como transacciones....

El uso de procedimientos almacenados para dar solución a este tipo de aplicaciones nos va a proporcionar:

* Más rapidez, ya que las órdenes SQL que conforman el procedimientos están pre-compiladas y el gestor no tiene que verificar que estén bien escritas (como tiene que hacer si empleamos las órdenes SQL directamente).
* Más simplicidad en la seguridad, ya que podemos especificar quien tiene permiso para ejecutar el procedimiento. Si usamos las órdenes SQL directamente tenemos que establecer la seguridad a nivel de tabla-columna (quien tiene permiso para realizar qué operación y con qué columnas) por lo que es más complejo.
* Más seguridad. El usar procedimientos con parámetros, nos va a permitir evitar acciones que comprometan la seguridad de la base de datos, como la inyección de código en formularios
* Estándar: En el sentido que existe un único lugar centralizado (el gestor de base de datos) donde se encuentran todos los procedimientos almacenados (asociados a una base de datos) y por lo tanto todos los programadores deberán de hacer uso de los procedimientos creados evitando duplicidades de código.
* Velocidad: Como el código ya se encuentra todo en el lado del servidor, no es necesario que los clientes envíen todas las órdenes, solamente los datos que van a necesitar los procedimientos para su funcionamiento.

Operaciones

**Crear procedimientos**

La orden SQL que permite crear un procedimiento almacenado es [CREATE PROCEDURE](https://dev.mysql.com/doc/refman/8.0/en/create-procedure.html).

[Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente](https://wiki.cifprodolfoucha.es/index.php?title=Archivo:Mod_BD_Prog_Proc_15.jpg)

Imagen obtenida de [https://dev.mysql.com](https://dev.mysql.com/doc/refman/8.0/en/create-procedure.html)

El conjunto de instrucciones van en el 'routine\_body'. Si lo forman varias (lo normal) deben ir entre las etiquetas BEGIN y END.

Fijarse que el END debe llevar punto y coma si a continuación le siguen nuevas instrucciones.

Veremos poco a poco las opciones que nos permite utilizar la orden SQL anterior.

Como mínimo debemos de indicar un nombre para el procedimiento y un conjunto de órdenes SQL entre las palabras clave BEGIN y END.

USE BD;

**DELIMITER** $$

**CREATE** **PROCEDURE** nombreProc()

**BEGIN**

*-- Instrucciones SQL a ejecutar dentro del procedimiento*

**END**$$

**DELIMITER** ;

Un procedimiento se crea dentro de una base de datos concreta.

Si no indicamos la base de datos, lo hará en la base de datos **activa** que recordar que en MysqlWorkBench es la que se visualiza en negrilla.

Cuando ejecutemos la orden CREATE PROCEDURE, podemos cambiar antes la base de datos activa con la orden SQL: USE nombre\_bd;

Otra forma de indicar en qué base de datos debe crearse el procedimiento es anteponiendo al nombre del mismo, el nombre de la base de datos seguido por un punto, de la forma: CREATE PROCEDURE **nombreBD.**nombreProc()

Entre las cláusulas que podemos emplear dentro del CREATE PROCEDURE estaría COMMENT que nos sirve para documentar cual es el objeto del procedimiento:

USE BD;

**DELIMITER** $$

**CREATE** **PROCEDURE** nombreProc()

**COMMENT** 'Objetivo del procedimiento'

**BEGIN**

*-- Instrucciones SQL a ejecutar dentro del procedimiento*

**END**$$

**DELIMITER** ;

Lo primero que llama la atención cuando se ve la definición de un procedimiento almacenado en Mysql es la orden SQL DELIMITER:

USE CIRCO;

**DELIMITER** $$

**CREATE** **PROCEDURE** animales\_getList()

**BEGIN**

**SELECT** nombre

**FROM** ANIMALES

**ORDER** **BY** nombre;

**END**$$

**DELIMITER** ;

Mysql toma como delimitador de sentencias SQL el punto y coma por defecto. Quiere decir que cada vez que encuentra un punto y coma intenta ejecutar la orden SQL.

Debemos de ver al procedimiento almacenado como un todo, como si fuera una única instrucción que queremos que Mysql ejecute entera.

Si mantenemos el punto y coma tendríamos algo así:

use employees;

**CREATE** **PROCEDURE** animales\_getList()

**BEGIN**

**END**;

¿ Pero qué pasa ahora si queremos poner una orden SQL dentro del procedimiento almacenado ?

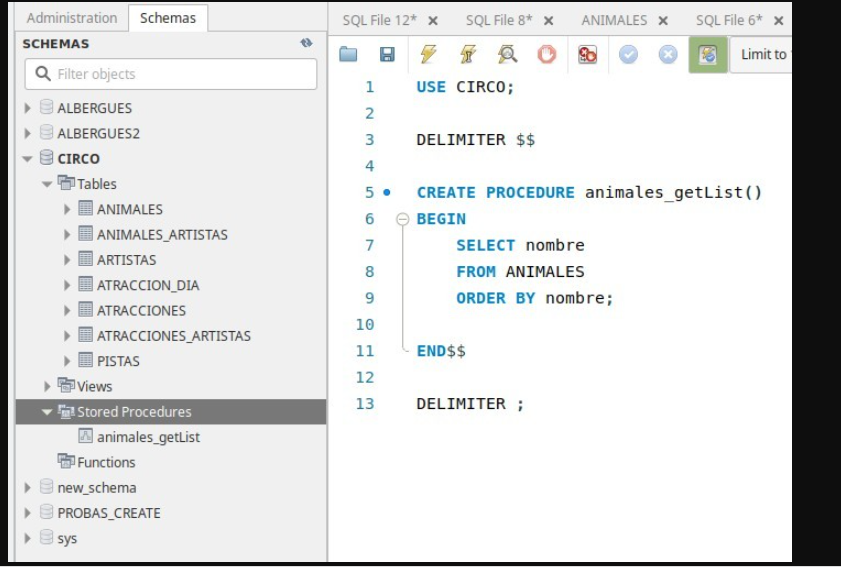
Que intentará ejecutarla y no la interpretará como parte del procedimiento almacenado.

Para evitarlo, indicamos con la orden SQL DELIMITER cual es el símbolo que debe encontrar Mysql para ejecutar la orden SQL (en nuestro caso el CREATE PROCEDURE) y por lo tanto dentro del mismo ya podemos poner sentencias SQL acabadas en punto y coma para separar unas de otras.

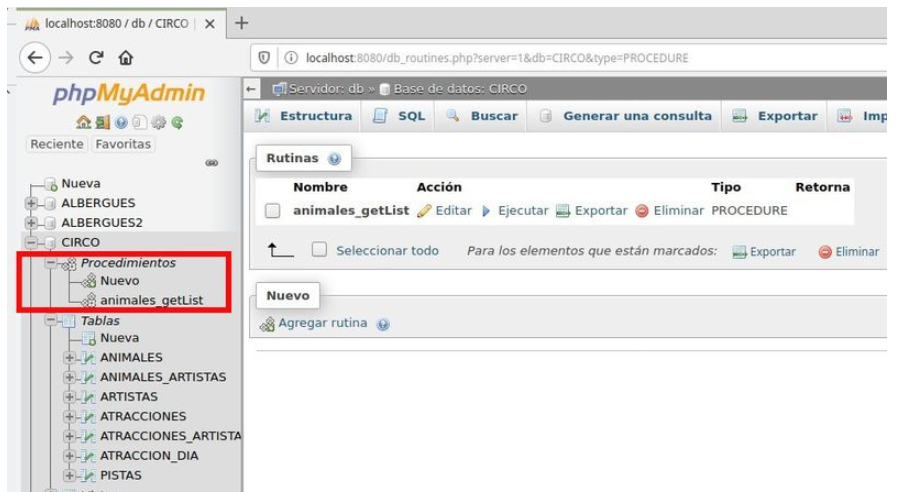
Cuando acabamos de definir el procedimiento almacenado volvemos a indicar que el delimitador sea ';' por si queremos ejecutar a continuación otras instrucciones SQL.

Indicar que el delimitador no tiene por qué ser $$, podría ser cualquiera que sepamos que no se va a emplear dentro del procedimiento almacenado, como por ejemplo %%.

Una vez creado, aparecerá en la sección **Stores Procedures** en MysqlWorkBench.

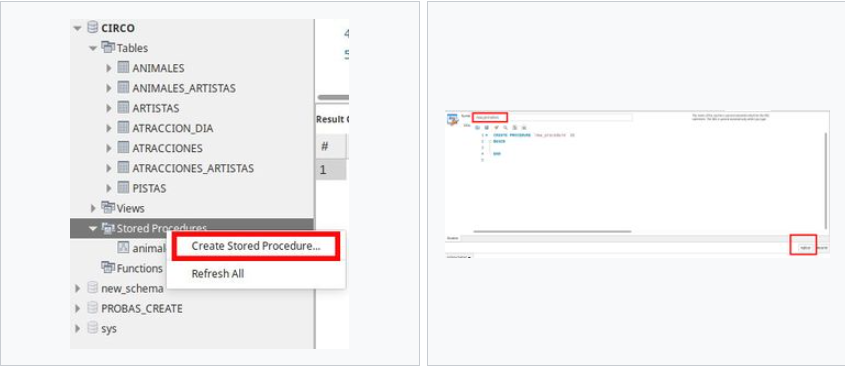


En Mysql disponemos de una sección específica para crear, borrar, modificar, ejecutar o exportar un procedimiento almacenado:



En MysqlWorkBench también podemos crearlo de la siguiente forma:

**Creando proc. almacenados en MysqlWorkBench**



Pulsando el botón derecho sobre la sección 'Stored Procedures**.**

Escribimos el nombre del procedimiento y el código que va a llevar. Al acabar pulsamos el botón Apply.

En este caso no hace falta poner DELIMITER, ya que se encargar el propio MysqlWorkBench de hacerlo...

**Listar procedimientos**

La orden SQL que permite ver la información de los procedimientos almacenados es: [SHOW PROCEDURE STATUS](https://dev.mysql.com/doc/refman/5.5/en/show-procedure-status.html).

Por ejemplo, para ver la información de los procedimientos almacenados de la base de datos 'CIRCO':

**SHOW** **PROCEDURE** STATUS

**WHERE** Db = 'CIRCO';

**Llamar a un procedimiento**

Para llamar a un procedimiento almacenado debemos hacer uso de la orden SQL [CALL](https://dev.mysql.com/doc/refman/5.7/en/call.html).

Al llamar a un procedimiento almacenado, las órdenes de su interior será ejecutadas.

Por ejemplo:

**CALL** CIRCO.animales\_getList();

Fijarse que delante del nombre del procedimiento almacenado lleva el nombre de la base de datos. Esto no sería necesario si estamos ejecutando el procedimiento con la base de datos seleccionada o teniendo la orden SQL: USE db\_name; antes de la llamada al procedimiento.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Visualizar el código de un procedimiento**

Para ver el código de un procedimiento almacenado podemos hacer uso de la orden SQL [SHOW CREATE PROCEDURE nombre\_proc](https://dev.mysql.com/doc/refman/5.7/en/show-create-procedure.html).

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**SHOW** **CREATE** **PROCEDURE** animales\_getList;

**Modificar un procedimiento**

En MYSQL no se permite modificar los parámetros y el cuerpo de un procedimiento.

Para ello tendremos que borrarlo y volverlo a crear. Lo que sí permite es modificar ciertas características del mismo. La orden SQL para modificar estas características es [ALTER PROCEDURE](https://dev.mysql.com/doc/refman/8.0/en/alter-procedure.html).Por ejemplo, podemos modificar la cláusula COMMNENT que sirve para tener una descripción del procedimiento.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**ALTER** **PROCEDURE** animales\_getList

**COMMENT** 'Obtiene un listado de todos los animales';

**SHOW** **PROCEDURE** STATUS

**WHERE** Db = 'CIRCO';

[Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente](https://wiki.cifprodolfoucha.es/index.php?title=Archivo:Mod_BD_Prog_Proc_6.jpg)

Desde el MysqlWorkBench podemos 'modificar' un procedimiento (realmente lo va a borrar y volverá a crearlo) de la forma:

**Borrar un procedimiento**

La orden SQL que permite borrar un procedimiento es [DROP PROCEDURE](https://dev.mysql.com/doc/refman/8.0/en/drop-procedure.html).

Indicar que podemos incluir una cláusula 'IF EXISTS' para indicar que solamente en caso de que exista, borre el procedimiento. Esto es muy útil cuando tenemos un bloque de sentencias, ya que si no ponemos el IF, si el procedimiento no existe daría un error y se dejarían de ejecutar las sentencias.

**DROP** **PROCEDURE** **IF** **EXISTS** nombre\_proc;

Indicar que podemos hacer referencia al procedimiento anteponiendo a su nombre el nombre de la base de datos donde está creado seguido del un punto:

DROP PROCEDURE IF EXISTS **CIRCO**.nombre\_proc;

Nota: Al igual que en la operación anterior, un procedimiento se puede borrar gráficamente desde el MysqlWorkBench (pulsando el botón derecho sobre el mismo) o desde el PhpMyadmin.

Ejercicios propuestos

* Probar a crear los procedimientos de diferentes formas: PhpMyAdmin, directamente en una pestaña SQL o a través del asistente de MysqlWorkBench.
* Después de crear el procedimiento llámalo.
* Crea cada procedimiento con un pequeño comentario de su objetivo.
* En caso de que no se indique ordenación, establecer vosotros mismos un criterio de ordenación.

### Ejercicio 1

Crea un procedimiento de nombre artistas\_getList() que devuelva el nombre y apellidos de los artistas separados por coma con el formato: apellidos,nombre ordenados de forma descendente.

### Ejercicio 2

Crea un procedimiento de nombre artistas\_getListAnimales() que devuelva los nombres de los artistas junto con su nif así como el nombre y peso de los animales que están atendidos por los artistas, ordenados por nif del artista y nombre del animal.

### Ejercicio 3

Crea un procedimiento de nombre atracciones\_getListConAntiguedad5() que devuelva los datos de las atracciones que hayan comenzado hace 5 años con respecto a la fecha actual. Tendrás que hacer uso de [alguna de las funciones Date Time](https://dev.mysql.com/doc/refman/8.0/en/date-and-time-functions.html). Intenta averiguar cual.

Fijarse que este procedimiento es un buen candidato para emplear un parámetro en donde indiquemos el número de años. Lo veremos después cuando expliquemos el paso de datos por parámetros.

### Ejercicio 4

Crea un procedimiento de nombre animales\_Leo\_getPista() que muestre los datos de la pista donde trabaja el animal de nombre 'Leo'. Hacerlo empleando una variable local que guarde el nombre de la pista. Después consultar los datos de la pista empleando dicha variable local.

Fijarse que este procedimiento es un buen candidato para emplear un parámetro en donde indiquemos el nombre del animal. Lo veremos después cuando expliquemos el paso de datos por parámetros.