**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.

**Ejercicio 5**

Crea un procedimiento de nombre atracciones\_getUltima() que obtenga los datos de la última atracción celebrada (tabla ATRACCION\_DIA), empleando variables locales. Para ello guarda en una variable el nombre de la última atracción celebrada y busca los datos de dicha atracción. Ten en cuenta limitar con LIMIT el número de filas que devuelva una consulta si no sabes con certeza que vaya a devolver una única fila y vas a guardar el datos en una variable.

**Ejercicio 6**

Crea un procedimiento de nombre atracciones\_getArtistaUltima() que obtenga los datos de la atracción y del artista que trabaja en dicha atracción, cuya fecha de inicio ha empezado más tarde. Emplea dos variables. Una para guardar el nif del artista y otra para guardar el nombre de la atracción.

Solución Ejercicios propuestos

**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.

USE CIRCO;

**DROP** **PROCEDURE** **IF** **EXISTS** artistas\_getList;

**DELIMITER** $$

**CREATE** **PROCEDURE** artistas\_getList()

**COMMENT** 'Devuelve el nombre completo de todos los artistas ordenado descendentemente'

**BEGIN**

**SELECT** CONCAT(apellidos,',',nombre) **as** nombreCompleto

**FROM** ARTISTAS

**ORDER** **BY** nombreCompleto **DESC**;

**END**$$

**DELIMITER** ;

**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.

USE CIRCO;

**DROP** **PROCEDURE** **IF** **EXISTS** artistas\_getListAnimales;

**DELIMITER** $$

**CREATE** **PROCEDURE** artistas\_getListAnimales()

**COMMENT** 'Devuelve el nombre, apellidos y nif de los artistas junto con el nombre y peso de los animales que atienden'

**BEGIN**

**SELECT** nif, apellidos, ARTISTAS.nombre **as** nombreArtista, ANIMALES.nombre **as** nombreAnimal, peso

**FROM** ARTISTAS **INNER** **JOIN** ANIMALES\_ARTISTAS **ON** (ANIMALES\_ARTISTAS.nif\_artista=ARTISTAS.nif)

**INNER** **JOIN** ANIMALES **ON** (ANIMALES\_ARTISTAS.nombre\_animal=ANIMALES.nombre)

**ORDER** **BY** nif, nombreAnimal;

**END**$$

**DELIMITER** ;

**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.

La función a emplear es [DATE\_SUB](https://dev.mysql.com/doc/refman/8.0/en/date-and-time-functions.html#function_date-sub) que tiene la misma sintaxis que [DATE\_ADD](https://dev.mysql.com/doc/refman/8.0/en/date-and-time-functions.html#function_date-add).

USE CIRCO;

**DROP** **PROCEDURE** **IF** **EXISTS** atracciones\_getListConAntiguedad5;

**DELIMITER** $$

**CREATE** **PROCEDURE** atracciones\_getListConAntiguedad5()

**COMMENT** 'Devuelve todos los datos de las atracciones que comenzaran hace 5 años con respecto a la fecha actual'

**BEGIN**

**SELECT** \*

**FROM** ATRACCIONES

**WHERE** fecha\_inicio **BETWEEN** DATE\_SUB(curdate(), INTERVAL 5 **YEAR**) **AND** curdate()

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

**END**$$

**DELIMITER** ;

**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.

USE CIRCO;

**DROP** **PROCEDURE** **IF** **EXISTS** animales\_Leo\_getPista;

**DELIMITER** $$

**CREATE** **PROCEDURE** animales\_Leo\_getPista()

**COMMENT** 'Devuelve los datos de la pista donde trabaja el animal de nombre Leo'

**BEGIN**

**DECLARE** v\_nombrePista varchar(50) **default** ''; *-- El tipo y tamaño debe de coincidir con el declarado en la tabla. Si no podemos nada, el valor por defecto es NULL*

**SELECT** nombre\_pista

**INTO** v\_nombrePista

**FROM** ANIMALES

**WHERE** nombre = 'Leo';

**SELECT** \* *-- Devuelve una única fila. No hace falta order by*

**FROM** PISTAS

**WHERE** nombre=v\_nombrePista;

**END**$$

**DELIMITER** ;

**Línea 8:** Fijarse que está declarada como varchar(50) que se corresponde al tipo de dato y tamaño de la columna nombre de la tabla PISTAS.

Se le da un valor por defecto. Esto es opcional, pero tenéis que tener en cuenta que si la orden SELECT no encuentra dato, no se realiza la parte INTO y por tanto a la salida de la orden SELECT la variable tendrá el valor por defecto. Cuando veremos la sentencia IF veremos que usaremos el valor por defecto para comprobar si queremos que siga ejecutando el procedimiento o salga del mismo.

**Ejercicio 5**

Crea un procedimiento de nombre atracciones\_getUltima() que obtenga los datos de la última atracción celebrada (tabla ATRACCION\_DIA), empleando variables locales. Para ello guarda en una variable el nombre de la última atracción celebrada y busca los datos de dicha atracción.

USE CIRCO;

**DROP** **PROCEDURE** **IF** **EXISTS** atracciones\_getUltima;

**DELIMITER** $$

**CREATE** **PROCEDURE** atracciones\_getUltima()

**COMMENT** 'Devuelve los datos de la última atracción celebrada'

**BEGIN**

**DECLARE** v\_nombreAtraccion varchar(50) **default** ''; *-- El tipo y tamaño debe de coincidir con el declarado en la tabla. Si no podemos nada, el valor por defecto es NULL*

**SELECT** nombre\_atraccion

**INTO** v\_nombreAtraccion

**FROM** ATRACCION\_DIA

**ORDER** **BY** fecha **DESC**

**LIMIT** 1;

*/\**

*Si empleáramos subconsultas, podríamos hacer algo como lo siguiente*

*SELECT nombre\_atraccion*

*INTO v\_nombreAtraccion*

*FROM ATRACCION\_DIA*

*WHERE fecha = (SELECT MAX(fecha)*

*FROM ATRACCION\_DIA)*

*LIMIT 1;*

*\*/*

**SELECT** \* *-- Devuelve una única fila. No hace falta order by*

**FROM** ATRACCIONES

**WHERE** nombre=v\_nombreAtraccion;

**END**$$

**DELIMITER** ;

**Ejercicio 6**

Crea un procedimiento de nombre atracciones\_getArtistaUltima() que obtenga los datos de la atracción y del artista que trabaja en dicha atracción, cuya fecha de inicio ha empezado más tarde. Emplea dos variables. Una para guardar el nif del artista y otra para guardar el nombre de la atracción.

USE CIRCO;

**DROP** **PROCEDURE** **IF** **EXISTS** atracciones\_getArtistaUltima;

**DELIMITER** $$

**CREATE** **PROCEDURE** atracciones\_getArtistaUltima()

**COMMENT** 'Devuelve los datos de la atracción y del artista que trabaja en dicha atracción, cuya fecha de inicio ha empezado más tarde'

**BEGIN**

**DECLARE** v\_nombreAtraccion varchar(50) **default** ''; *-- El tipo y tamaño debe de coincidir con el declarado en la tabla. Si no podemos nada, el valor por defecto es NULL*

**DECLARE** v\_nifArtista char(9); *-- Al no llevar default el valor por defecto es null y sería lo que tendríamos que comparar en un if por ejemplo.*

**SELECT** nombre\_atraccion,nif\_artista

**INTO** v\_nombreAtraccion,v\_nifArtista

**FROM** ATRACCIONES\_ARTISTAS

**ORDER** **BY** fecha\_inicio **DESC**

**LIMIT** 1;

**SELECT** \* *-- Devuelve una única fila. No hace falta order by*

**FROM** ATRACCIONES,ARTISTAS

**WHERE** ATRACCIONES.nombre=v\_nombreAtraccion **AND**

ARTISTAS.nif = v\_nifArtista;

**END**$$

**DELIMITER** ;