A4. Creación y depuración de Procedimientos Almacenados.

Índice.

A4. Creación y depuración de Procedimientos Almacenados.

1. ¿Qué es un Procedimiento Almacenado?

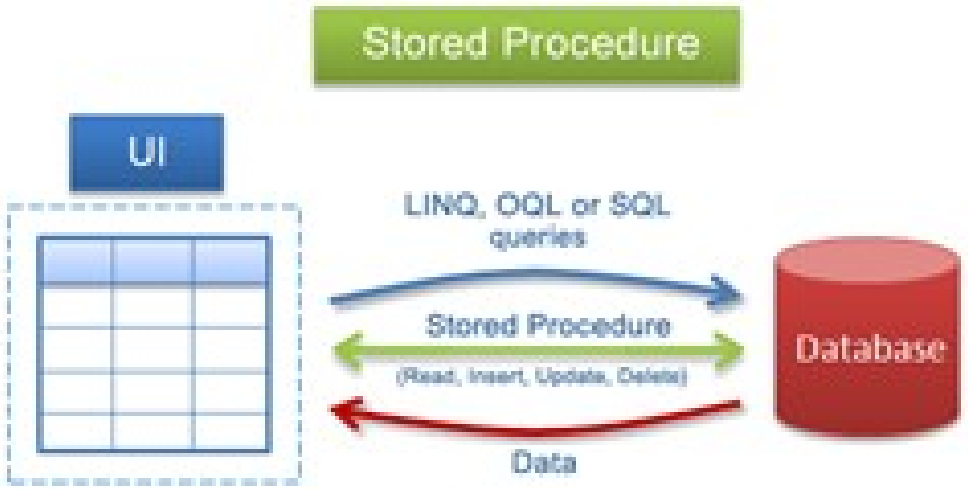
A4. Creación y depuración de Procedimientos Almacenados.

1. ¿Qué es un Procedimiento Almacenado?

Un **procedimiento almacenado** es un conjunto de instrucciones SQL que se almacena asociado a una base de datos.

- Creación → CREATE PROCEDURE
- Invocación → CALL

El procedimiento almacenado puede tener cero o muchos parámetros, que pueden ser de entrada, salida o de entrada y salida.



2. Sintaxis.

La sintaxis para la creación de un Procedimiento Almacenamiento es la siguiente:

```
CREATE
    [DEFINER = { user | CURRENT_USER }]
    PROCEDURE sp_name ([proc_parameter[,...]])
    [characteristic ...] routine_body

proc_parameter:
    [ IN | OUT | INOUT ] param_name type

func_parameter:
    param_name type

type:
    Any valid MySQL data type

characteristic:
    COMMENT 'string'
  | LANGUAGE SQL
  | [NOT] DETERMINISTIC
  | { CONTAINS SQL | NO SQL | READS SQL DATA | MODIFIES SQL DATA }
  | SQL SECURITY { DEFINER | INVOKER }

routine_body:
    Valid SQL routine statement
```

3. Componentes.

Algunos componentes importantes son los siguientes:

- **Delimitadores** → la definición de un procedimiento almacenado necesita, temporalmente, modificar el carácter separador de las sentencias SQL que, por defecto, es ';'.

  Modificación del delimitador a '//': `DELIMITER //`

  Finalización del área: `// DELIMITER ;`

- **Parámetros de entrada, salida y entrada/salida** → los procedimientos almacenados pueden necesitar datos que les sean comunicados mediante parámetros, emitir resultados sobre dichos parámetros, o, modificar el estado de algún parámetro.

```
Create Procedure mostrarConcellos( IN Provincia varchar( 20 ), INOUT Cantidad int, OUT Resultado varchar( 250 ) )
```

  Parámetro de entrada IN: Provincia

  Parámetro de entrada/salida INOUT: Cantidad

  Parámetro de salida OUT: Resultado

4. Asignación de resultados a los parámetros de salida.

La asignación de valores a las variables de salida (o entrada/salida) puede realizarse de estas dos formas:

- SET variable =

```
DELIMITER $$
DROP PROCEDURE IF EXISTS contar_productos$$
CREATE PROCEDURE contar_productos(IN gama VARCHAR(50), OUT total INT UNSIGNED)
BEGIN
  SET total = (
    SELECT COUNT(*)
    FROM producto
    WHERE producto.gama = gama);
END
$$
```

- SELECT … INTO variable

```
DELIMITER $$
DROP PROCEDURE IF EXISTS contar_productos$$
CREATE PROCEDURE contar_productos(IN gama VARCHAR(50), OUT total INT UNSIGNED)
BEGIN
  SELECT COUNT(*)
  INTO total
  FROM producto
  WHERE producto.gama = gama;
END
$$
```

5. Invocación de los Procedimientos Almacenados.

La invocación (o llamada) a los procedimientos almacenados se realiza con la palabra reservada CALL:

```
CALL contar_productos('Herramientas', @total);
SELECT @total;
```

En este ejemplo, se llama al procedimiento almacenado contar_productos pasando como parámetro de entrada '**Herramientas**', que es la gama de herramientas que se desea contar, y su número se devuelve sobre la variable de usuario **total**, cuyo valor posteriormente se muestra a través de la función SELECT.

6. Eliminación de un Procedimiento Almacenado.

La eliminación de un procedimiento almacenado se realiza a través del comando DROP:

```
DROP PROCEDURE [IF EXISTS] sp_name
```

```
DELIMITER $$
DROP PROCEDURE IF EXISTS contar_productos$$
```