Procedimientos. Herramientas Complementarias y Sintaxis

1. Herramientas Complementarias
2. Sintaxis

Herramientas Complementarias

DELIMITER

Declaramos un DELIMITER cuando sea necesario introducir conjuntos de sentencias que se tienen que ejecutar como una sola.

Podemos cambiar el delimitador por defecto (;) si así se lo indicamos al sistema gestor.

Aunque cambiemos el delimitador, el sistema gestor sigue aceptando el punto-y-coma como final de instrucción.

```
DELIMITER ;
DELIMITER $$
```

VARIABLES

En SQL podemos encontrar dos tipos de variables:

- Variables declaradas: Son en las que predefinimos un tipo de datos a contener antes de ser usadas.

```
DECLARE var1 VARCHAR(20);
DECLARE var2 INTEGER DEFAULT 0;
DECLARE var3 INTEGER;
```

- Variables no declaradas: Son las que no tienen ninguna declaración previa ni tipo de datos asignado.

```
@aux;
@cadena_texto;
```

ASIGNACIÓN DE VALORES EN VARIABLES

Utilizaremos la herramienta SET para modificar los valores guardados en las variables.

- Variables declaradas:

```
SET var1 = 'Hola';
SET var2 = var2+1;
SET var3 = (SELECT COUNT(id_usuario) FROM usuario);
```

- Variables no declaradas:

```
SET @aux = TRUE;
SET @cadena_texto = 'Adiós';
```

SENTENCIAS DINÁMICAS

Si deseamos construir sentencias dinámicas utilizaremos la estructura siguiente:

```
SET @aux = CONCAT("INSERT INTO ", nombre_tabla, " SELECT * FROM " ,
nombre_bbdd, ".", nombre_tabla);
PREPARE stmt1 FROM @aux;
EXECUTE stmt1;
DEALLOCATE PREPARE stmt1;
```

Es muy útil mara construir sentencias con partes que se tienen que evaluar des de una variable.

Sintaxis

PARÁMETROS DE LOS PROCEDIMIENTOS

En la cabecera de los procedimientos podemos tener tres tipos de parámetros. IN (entrada), OUT (salida) y INOUT (entrada y salida).

Se definen de la manera siguiente:

<IN/OUT/INOUT> <nombre_parametro> <tipo_parametro>

```
IN vid INTEGER
OUT vfecha DATE
INOUT vcadena VARCHAR(50)
```

IMPORTANTE

Hay que tener en cuenta que tanto los nombres de los parámetros como de las variables no sean iguales a los atributos de ninguna de las tablas usadas en el procedimiento, ya que esto podría inducir a errores a la hora de ejecutar el código

La estructura típica a la hora de trabajar con procedimientos es usando delimitadores, borrando el procedimiento y posteriormente crearlo.

```
DELIMITER $$
DROP PROCEDURE IF EXISTS nombre_procedure $$
CREATE PROCEDURE nombre_procedure
    ({IN/OUT/INOUT} nombre_parametro tipo_parametro)
BEGIN
    // Cuerpo del procedimiento: Variables, condicionales, bucles,
    selecciones, etc.
END $$
DELIMITER ;
```

EJECUCIÓN DE LOS PROCEDIMIENTOS

Para llamar a un procedimiento se usa la cláusula CALL.

Podemos llamar a procedimientos que no tengan parámetros:

```
CALL procedimiento1();
```

Con parámetros de entrada:

```
CALL procedimiento2("texto",1,@aux);
```

O con parámetros de salida:

```
CALL procedimiento3(@aux);
```

“Vive como si fueras a morir mañana. Aprende como si fueras a vivir siempre“

MAHATMA GANDHI