TEMA 5. PROGRAMACIÓN DE BASES DE DATOS
(Parte I - Procedimientos Almacenados)

1. PROCEDIMIENTOS ALMACENADOS EN Microsoft SQL Server

2. DESENCADENADORES O TRIGGERS EN Microsoft SQL Server

1. PROCEDIMIENTOS ALMACENADOS EN Microsoft SQL Server

DEFINICIÓN

- Los procedimientos almacenados son una serie de **sentencias Transact-SQL que son posible llamarlos mediante un identificador** y **pueden recibir** y **devolver valores**.
- Hay **procedimientos almacenados que vienen definidos por el propio sistema** (procedimientos almacenados de sistema) y **otros que los define el propio usuario** (procedimientos almacenados de usuario).
- Dentro de un procedimiento almacenado se puede:
  - Definir variables: DECLARE @NOMBRE TIPO
  - Utilizar estructuras de control de flujo.
  - Llamar a otro procedimiento almacenado.
  - Devuelven un valor de estado (status value) comprendidos entre el 0 y -14, por ejemplo el valor 0 es ejecución con éxito.

(Ver " lenguaje de control de flujo" en enlace Manual de Referencia de Transact-SQL)

PRINCIPALES VENTAJA

- Se **optimizan** en el momento de su creación, por lo tanto mejora el rendimiento de las consultas.
- Están **compilados y almacenados en el servidor** para que su ejecución sea más rápida, que el conjunto de sentencias individuales que lo integran.

CREACIÓN

La sentencia Transact-SQL que permite crear procedimientos almacenados es CREATE PROCEDURE. Una sintaxis reducida es la siguiente:

CREATE PROCEDURE [propietario.]nombre_procedimiento [
{@*parámetro tipoDatos* }[**=** *predeterminado*] [OUTPUT]] [, …n]

AS

[BEGIN]
instrucciónSQL [...*n*]
[RETURN N]

[END]

Parámetros

[{@*parámetro tipoDatos* }[**=** *predeterminado*] [OUTPUT]]

Ejemplo:

CREATE PROCEDURE calcular @p1 int=1, @p2 char, @p3 varchar(8)='Tomas' AS ...

En la llamada existen muchas posibilidades, veamos algunas:

EXEC calcular @p2='A'

EXEC calcular 3,'B'

EXEC calcular 3, 'B', 'JAVIER'

EXEC calcular @p1=DEFAULT, @p2='D'

Retorno de información

Los procedimientos almacenados pueden retornar información al procedimiento que los llamó o al usuario de **tres modos** diferentes:

- ✔ **Mediante el retorno de un valor de estado**.
- ✔ **Mediante parámetros de retorno o de salida** (Cláusula **OUTPUT**).
- ✔ Mediante la cláusula **RETURN**.

EJECUCIÓN

[EXEC[UTE]]
[@*valor de retorno* **=**] *nombreProcedimiento*
[[@*parámetro* **=**] {*valor* | @*variable* [OUTPUT] | [DEFAULT] }] [**,**…*n*]

EXEC o EXECUTE es necesario en los siguientes casos:

- ✔ Si llamamos al procedimiento desde otro procedimiento almacenado, desde un trigger, desde el código de un programa donde la llamada se encuentre inmersa.
- ✔ Si queremos obtener un valor de estado.
- ✔ Si tiene parámetro de salida.
- ✔ Si hemos utilizado la cláusula RETURN.

Ejemplos:

Creación del procedimiento:

```
CREATE PROCEDURE INFOEMPLEADOS AS

    SELECT * FROM TEMPLE
```

Ejecución del procedimiento:

```
EXEC INFOEMPLEADOS
```

Borrado de procedimientos almacenados

```
    DROP PROCEDURE nombre_procedimiento
```

Cambiar nombre a un procedimiento almacenado

```
    sp_rename nombre_antiguo, nombre_nuevo
```

Aquí utilizamos un procedimiento almacenado (store procedure) del sistema.

Veamos un ejemplo de **procedimiento almacenado simple** que se encuentra en "Ej1 pa simple.sql". El archivo pertenece a: EJEMPLOS_PROCEDIMIENTOS_ALMACENADOS.

```
--Creación de un procedimiento almacenado sencillo.

/*Deseamos conocer para todos los empleados su número, nombre y el
nombre del departamento en el que se encuentra.*/
USE empresa;

CREATE PROCEDURE p1
AS
BEGIN

SELECT  numem,nomem,nomde
FROM tdepto d  RIGHT JOIN temple e ON (e.numde=d.numde)

END;

--EJECUCIÓN.
p1;

EXEC p1;

EXECUTE p1;

--Podríamos consultar el valor de estado.
--Podremos ver el valor devuelto en la pestaña Messages

DECLARE @valor INT
EXECUTE @valor= p1
PRINT @valor
```

Veamos unos ejemplos de **procedimientos almacenados con un parámetro de entrada** que se encuentran en "Ej2 pa con parametro de entrada.sql". El archivo pertenece a: EJEMPLOS_PROCEDIMIENTOS_ALMACENADOS.

```
/*Creación de un procedimiento almacenado con un parámetro de
entrada.*/

/*Deseamos saber cuáles son los empleados de un departamento que se
pasa como parámetro*/
USE empresa;

CREATE PROCEDURE p2 @NOMBREDEPARTAMENTO VARCHAR(50)
AS
BEGIN

SELECT numem,nomem
FROM tdepto d JOIN  temple e ON (d.numde=e.numde)
WHERE  d.nomde = @NOMBREDEPARTAMENTO

END;

--EJECUCIÓN
p2 'NOMINAS';

EXECUTE p2 'SECTOR SERVICIOS';

EXEC p2 @NOMBREDEPARTAMENTO='PERSONAL';

-- Así nos daría error, pues espera un parámetro de entrada
p2;

DROP PROCEDURE p2;
```

```
/*Creación de un procedimiento almacenado con un parámetro de entrada
al que se le asigna un valor por defecto.*/
CREATE OR ALTER PROCEDURE p2 @NOMBREDEPARTAMENTO VARCHAR(50)='SECTOR
SERVICIOS'
AS
BEGIN

SELECT  numem,nomem
FROM tdepto d JOIN temple e ON ( d.numde=e.numde)
WHERE d.nomde = @NOMBREDEPARTAMENTO

END;

--EJECUCIÓN.
p2 'NOMINAS';

p2 @NOMBREDEPARTAMENTO='PERSONAL';

p2;

p2 @NOMBREDEPARTAMENTO=DEFAULT;

EXEC p2 DEFAULT;

--Podríamos consultar el valor de estado.
DECLARE @valor INT;
EXECUTE @valor= p2;
PRINT @valor;
```

Veamos unos ejemplos de **procedimientos almacenados con parámetros de entrada y salida** que se encuentran en “Ej3 pa con parametro de entrada y salida.sql”. El archivo pertenece a: EJEMPLOS_PROCEDIMIENTOS_ALMACENADOS.

```
/*Creación de un procedimiento almacenado con un parámetro de entrada y
un parámetro de salida.*/

/*Deseamos saber cuántos empleados tiene un departamento que se pasa
como parámetro.*/
USE empresa;

CREATE PROCEDURE p3 @NOMBREDEPARTAMENTO VARCHAR(50), @NUMERO INT OUTPUT
AS
BEGIN

SELECT @NUMERO = COUNT(*)
FROM temple e JOIN tdepto D ON (e.numde=d.numde)
WHERE d.nomde LIKE @NOMBREDEPARTAMENTO

END;

--EJECUCIÓN.
DECLARE @NUM INT;
EXEC p3 'SECTOR SERVICIOS', @NUM OUTPUT;
PRINT @NUM;

DECLARE @NUM INT;
EXEC p3 'NOMINAS',@NUM OUTPUT;
PRINT @NUM;
```

```
/*Creación de un procedimiento almacenado con un parámetro de entrada y
dos parámetros de salida.*/

/*Deseamos saber cuántos empleados tiene un departamento que se
pasa como parámetro y la edad de su empleado más joven.*/
CREATE OR ALTER PROCEDURE p3v1 @NOMBREDEPARTAMENTO VARCHAR(50),
@NUM_EMP INT OUTPUT,@MIN_EDAD  INT  OUTPUT
AS
BEGIN

SELECT @NUM_EMP = COUNT(*),
       @MIN_EDAD=MIN(DATEDIFF(DAY,fecna,GETDATE())/365)
FROM temple e JOIN tdepto d ON (e.numde=d.numde)
WHERE d.nomde LIKE @NOMBREDEPARTAMENTO

END;

--EJECUCIÓN.
DECLARE @NUM INT, @EDAD INT;
EXEC p3v1 'SECTOR SERVICIOS', @NUM OUTPUT,@EDAD OUTPUT;
PRINT @NUM;
PRINT @EDAD;

--Podríamos consultar el valor de estado.
DECLARE @return_status INT;
DECLARE @NUM INT;
EXECUTE @return_status = p3 'NOMINAS',@NUM OUTPUT;
PRINT @return_status;
PRINT @NUM;
```

Veamos unos ejemplos de **procedimientos almacenados que devuelven un valor con la cláusula RETURN** que se encuentra en “Ej4 pa con return.sql”. El archivo pertenece a: EJEMPLOS_PROCEDIMIENTOS_ALMACENADOS.

```
/*Creación de un procedimiento almacenado que devuelve un valor con la
cláusula  RETURN.*/

--RETURN devuelve un valor entero.

/*Este procedimiento almacenado devuelve un 1 si la media de los
salarios de un departamento pasado por parámetro supera los 1800 euros,
en caso contrario devuelve un 2.*/
USE empresa;

CREATE PROCEDURE p4 @NOMBREDEPARTAMENTO VARCHAR(50)
AS
BEGIN

IF  (SELECT AVG(salar) FROM tdepto d JOIN temple e ON (d.numde=e.numde)
     WHERE nomde LIKE @NOMBREDEPARTAMENTO) > 1800
BEGIN
  PRINT 'MAYOR QUE 1800'
  RETURN 1
END
ELSE
BEGIN
  RETURN 2
END

END;

--EJECUCIÓN
DECLARE @VALOR INT;
EXEC @VALOR=p4 'NOMINAS';
PRINT @VALOR;

DECLARE @VALOR INT;
EXEC @VALOR=p4 'ORGANIZACION';
PRINT @VALOR;
```

```
/*Este procedimiento no da error en la creación, aunque devolvamos un
valor que no sea entero, sin embargo, observa que el valor devuelto lo
convierte a entero.*/
CREATE OR ALTER PROCEDURE p4v1 @NOMBREDEPARTAMENTO VARCHAR(50)
AS
BEGIN

DECLARE @MEDIA FLOAT

SELECT @MEDIA=AVG(salar) FROM tdepto d JOIN temple e ON
(d.numde=e.numde)
     WHERE nomde LIKE @NOMBREDEPARTAMENTO

PRINT 'LA MEDIA ES'

RETURN @MEDIA

END;

--EJECUCIÓN.
DECLARE @VALOR FLOAT;
EXEC @VALOR=P4V1 'ORGANIZACION';
PRINT @VALOR;

--Comprobación:
SELECT AVG(salar) FROM tdepto d JOIN temple e ON (d.numde=e.numde)
WHERE nomde LIKE 'ORGANIZACION';
```

Veamos un ejemplo de **procedimiento almacenado que llama a otro procedimiento almacenado** que se encuentra en "EJ5 pa que llama al pa del EJ4.sql". El archivo pertenece a: EJEMPLOS_PROCEDIMIENTOS_ALMACENADOS.

```
/*Este procedimiento almacenado llama al procedimiento almacenado p4, y
dependiendo del valor devuelto, muestra  un determinado mensaje.*/

USE empresa;

CREATE PROCEDURE p5 @NOMBREDEPARTAMENTO VARCHAR(50)
AS
BEGIN

DECLARE @VALOR_RETORNO INT
EXEC @VALOR_RETORNO = p4 @NOMBREDEPARTAMENTO

IF (@VALOR_RETORNO = 1)
BEGIN
      PRINT 'LA MEDIA DE LOS SALARIOS DEL DEPARTAMENTO ' +
@NOMBREDEPARTAMENTO + ' SUPERA LOS 1800 EUROS'
END
ELSE
BEGIN
      PRINT 'NO LA SUPERA'
END

END;

--Comprobaciones:
SELECT d.nomde, AVG(salar)
FROM tdepto d JOIN temple e ON (d.numde=e.numde)
GROUP BY d.nomde;

--EJECUCIÓN.
EXEC p5 'NOMINAS';

EXEC p5 'ORGANIZACION';
```

```
--Borrado de Procedimientos almacenados.
DROP PROCEDURE p1;
DROP PROCEDURE p2, p3, p4, p5;
DROP PROCEDURE p3v1, p4v1;
```