Taller	SQL SERVER PROGRAMACIÓN
Docente	Mg. Ing. Eric Gustavo Coronel Castillo
Tema	**INTRODUCCIÓN A LA PROGRAMACIÓN**

Bloque anónimo

Sintaxis

```
BEGIN

   Sentencias a ejecutar

END;
GO
```

Ejemplo 1

```
BEGIN

   DECLARE  @NUM1 INT,  @NUM2 INT,  @SUMA INT;

   SET @NUM1 =  CAST( RAND() * 100 AS INT );
   SET @NUM2 =  CAST( RAND() * 100 AS INT );

   SET @SUMA = @NUM1 + @NUM2;

   PRINT CONCAT( 'NUM1 = ', @NUM1 );
   PRINT CONCAT( 'NUM2 = ', @NUM2 );
   PRINT CONCAT( 'SUMA = ', @SUMA );

END;
GO
```

Funciones

Función Escalar

Sintaxis

```
CREATE FUNCTION [ esquema. ] nombre_funcion
(
    [      @parametro tipo_dato [ = valor_defecto ] [ READONLY ]
           [ ,...n ]
    ]
)
RETURNS tipo_dato_retorno
[ AS ]
BEGIN
    cuerpo_función
    RETURN expression;
END;
```

Ejemplo 2

```
CREATE FUNCTION dbo.fn_suma ( @num1 int, @num2 int )
RETURNS int
AS
BEGIN
    DECLARE @suma int;
    SET @suma = @num1 + @num2;
    RETURN @suma;
END;
GO

SELECT dbo.fn_suma( 24, 56 ) as suma;
GO

suma
-----------
80
```

Función de tabla en línea

Sintaxis

```
CREATE FUNCTION [ esquema. ] nombre_función
(
   [      @parametro tipo_dato [ = valor_defecto ] [ READONLY ]
          [ ,...n ]
   ]
)
RETURNS TABLE
[ AS ]
RETURN [ ( ] sentencia_select [ ) ] ;
```

Ejemplo 3

```
USE RH;
GO

CREATE FUNCTION dbo.fn_empleados ( @p_dpto int )
RETURNS TABLE
AS
RETURN
   SELECT idempleado, apellido, nombre
   FROM dbo.empleado
   WHERE iddepartamento = @p_dpto;
GO

SELECT * FROM dbo.fn_empleados(103);
GO
```

Función de tabla de múltiples instrucciones

Sintaxis

```
CREATE FUNCTION [ esquema. ] nombre_función
(
    [      @parametro tipo_dato [ = valor_defecto ] [ READONLY ]
           [ ,...n ]
    ]
)
RETURNS @return_variable TABLE definición_tipo_tabla
[ AS ]
BEGIN
    cuerpo_funcion
    RETURN;
END;
```

Ejemplo 4

```
CREATE FUNCTION dbo.fn_catalogo ( )
RETURNS @tabla TABLE
(
    codigo int identity(1,1) primary key not null,
    nombre varchar(50) not null,
    precio money not null
)
AS
BEGIN
    INSERT INTO @tabla(nombre,precio) values('Televisor', 1500.00);
    INSERT INTO @tabla(nombre,precio) values('Refrigeradora', 1450.00);
    INSERT INTO @tabla(nombre,precio) values('Lavadora', 1350.00);
    RETURN;
END;
GO

SELECT * FROM dbo.fn_catalogo();
GO
```

Ejemplo 5

```
USE RH;
GO

CREATE FUNCTION dbo.fn_planilla ( )
RETURNS @planilla TABLE
(
    codigo int primary key not null,
    nombre varchar(50) not null,
    plan_actual money not null,
    plan_proyectada money not null
)
AS
BEGIN
    INSERT INTO @planilla
    SELECT
           d.iddepartamento as codido,
           d.nombre as nombre,
           SUM(e.sueldo) as "planilla actual",
           cast(SUM(e.sueldo * 1.15) as money) as "planilla proyectada"
    FROM dbo.departamento as d
    JOIN dbo.empleado as e
    ON d.iddepartamento = e.iddepartamento
    GROUP BY d.iddepartamento, d.nombre;
    RETURN;
END;
GO

SELECT * FROM dbo.fn_planilla ( )
GO
```

Procedimientos

Sintaxis

```
CREATE { PROC | PROCEDURE } [ esquema.] nombre_procedimiento
    [ @parametro tipo_dato [ = default ] [ OUT | OUTPUT ] [READONLY] ]
    [ ,...n ]
AS
BEGIN

    cuerpo del procedimiento

END;
```

Ejemplo 6: Procedimiento simple

```
USE EDUCA;
GO

CREATE PROCEDURE dbo.usp_lista_cursos
AS
BEGIN
    SET NOCOUNT ON;
    SELECT * FROM dbo.curso;
END;
GO

EXEC dbo.usp_lista_cursos;
GO
```

Ejemplo 7: Procedimiento con parámetros

```
CREATE PROCEDURE dbo.usp_suma ( @num1 int, @num2 int )
AS
BEGIN
    DECLARE @suma int;
    SET @suma = @num1 + @num2;
    SELECT @num1 NUM1, @num2 NUM2, @suma SUMA;
END;
GO

EXEC dbo.usp_suma 54, 76;
GO
```

Ejemplo 8: Procedimiento con parámetro de salida

```
USE EDUCA;
GO

CREATE PROCEDURE dbo.usp_precio ( @p_idcurso int, @p_precio money OUT )
AS
BEGIN
    SELECT @p_precio = cur_precio
    FROM dbo.CURSO
    WHERE cur_id = @p_idcurso;
END;
GO

BEGIN
    DECLARE @precio money;
    EXEC dbo.usp_precio 3, @precio OUT;
    PRINT CONCAT( 'PRECIO: ', @precio );
END;
GO

PRECIO: 1815.00
```

Elementos de programación

Variables

Sintaxis:

```
DECLARE
   @nombre_variable [AS] data_type [ = value ] [ , ... ]
```

Sentencia de asignación

Instrucción SET

```
SET @nombre_variable = expresión;

SET @nombre_variable = ( sentencia_select );
```

Sentencia SELECT

```
SElECT @nombre_variable = ( sentencia SELECT ), . . .
FROM . . .
```

Ejercicios propuestos

1. Desarrolle una función que permita calcular el promedio de 3 números.
2. Desarrolle una función que reporte de la cantidad de alumnos matriculados y las vacantes disponibles de un determinado curso.
3. Desarrolle una función que dado el código de un curso reporte los alumnos que tienes saldo pendiente.
4. Desarrolle un procedimiento que reporte por cada curso la cantidad de alumnos matriculados, el importe proyectado y el importe recaudado.