Práctica 20

Transact SQL

Objetivo

*En esta práctica se introduce el concepto de programación de aplicaciones dentro de una base de datos, mediante el empleo del lenguaje **Transact SQL**.*

Introducción

Cuando se desea realizar una aplicación completa para el manejo de una base de datos relacional, resulta necesario utilizar alguna herramienta que soporte la capacidad de consulta del SQL y la versatilidad de los lenguajes de programación tradicionales.

Transact SQL es el lenguaje de programación que proporciona Microsoft SQL Server para extender el SQL estándar con otro tipo de instrucciones y elementos propios de los lenguajes de programación.

Con **Transact SQL** se pueden programar las unidades de programa de la base de datos **SQL Server**, estas son:

- Procedimientos almacenados
- Funciones
- Triggers
- Scripts

Transact SQL proporciona una variedad predefinida de tipos de datos, casi todos ellos son similares a los soportados por SQL.

Para declarar variables en **Transact SQL** se debe utilizar la palabra clave **declare**, seguido del nombre y tipo de datos de la variable, en **Transact SQL** los nombres de las variables deben comenzar con @, y no diferencia mayúsculas de minúsculas.

Transact SQL permite la creación de tipos de datos personalizados, con la instrucción **CREATE TYPE**.

Transact SQL proporciona el control de errores a través de las instrucciones **TRY** y **CATCH**, estas instrucciones suponen un gran apoyo para el control de flujo en

caso de un error. En el bloque **CATCH** están disponibles ciertas funciones especiales para la obtención de información detallada del error.

En ocasiones puede ser útil que se genere una excepción cuando los datos incumplen una regla de negocio, una excepción se puede generar en tiempo de ejecución a través de la función RAISERROR.

Equipo necesario

Una computadora con sistema operativo ***Windows*** que cuente con ***SQL Server 2022*** y ***SSMS***.

Metodología

1. En el menú de programas de Windows, localice la aplicación ***SQL Server Management Studio 20***.
2. En la ventana emergente, oprima el botón ***Connect***.
3. En la barra de herramientas localice y oprima el botón de ***New Query***.
4. En la barra de herramientas localice la lista desplegable ***Available Databases***, y seleccione ***empresa***.
5. En la ventana del editor de consultas transcriba el siguiente script con una estructura ***IF - ELSE***:

```
DECLARE @Ciudad varchar(20), @Nombre varchar(25)
SET @Nombre = 'Luis'
SET @Ciudad = 'Zamora'
IF EXISTS (SELECT * FROM Cliente WHERE Nombre = @Nombre)
BEGIN
  UPDATE Cliente SET Nombre='Luisito', Ciudad = @Ciudad WHERE
Nombre = @Nombre
END
ELSE
BEGIN
  SELECT * FROM Cliente
END
```

6. Oprima el botón de ***Execute***, para hacer la actualización del registro de Luis, y nuevamente oprima ***Execute*** para que en esta ocasión se realice la consulta.
7. Obtenga una imagen completa de la pantalla, mostrando el script y el resultado de la ejecución y consérvela como evidencia.
8. Ahora, oprima el botón de ***New Query*** y transcriba el siguiente script con una estructura ***CASE - WHEN***:

```
DECLARE @Ciudad varchar(20) = 'Morelia', @Nombre varchar(25)
SET @Nombre = (CASE
    WHEN @Ciudad = 'Uruapan' THEN (SELECT Nombre FROM Cliente
```

```
WHERE Ciudad
 = @Ciudad)
    WHEN @Ciudad = 'Morelia' THEN (SELECT Nombre FROM Cliente
WHERE Ciudad
= @Ciudad)
ELSE 'Inexistente'
END)
PRINT @Nombre
```

9. Oprima el botón de ***Execute***.
10. Obtenga una imagen completa de la pantalla, mostrando el script y el resultado de la ejecución y consérvela como evidencia.
11. Ahora, oprima el botón de ***New Query*** y transcriba el siguiente script, mostrando un bloque ***WHILE***, conteniendo sentencias CONTINUE y BREAK.

```
DECLARE @contador INT
SET @contador = 0
WHILE (@contador < 100)
BEGIN
   SET @contador = @contador + 1
   IF (@contador % 2 = 0)
     CONTINUE
   IF (@contador = 25)
    BREAK
   PRINT 'Iteracion:' + CAST(@contador AS varchar)
END
```

12. Oprima el botón de ***Execute***.
13. Obtenga una imagen completa de la pantalla, mostrando el script y el resultado de la ejecución y consérvela como evidencia.
14. Ahora, oprima el botón de ***New Query*** y transcriba el siguiente script, el cual contiene un bloque ***WHILE EXISTS***, para los resultados de una consulta ***SELECT***.

```
DECLARE @Num INT, @Cant INT
WHILE EXISTS (SELECT * FROM Pedido WHERE Cantidad > 0)
BEGIN
   SET @Num = (SELECT TOP 1 Número FROM Pedido WHERE Cantidad >
0)
UPDATE Pedido SET Cantidad = Cantidad - 1 WHERE Número = @Num
    SELECT @Cant = Cantidad FROM Pedido WHERE Número = @Num
  PRINT CONCAT('Pedido: ', @Num, '  Cantidad: ', @Cant)
END
```

15. Oprima el botón de ***Execute*** para disminuir las cantidades de todos los pedidos hasta cero.
16. Obtenga una imagen completa de la pantalla, mostrando el script y el resultado

de la ejecución y consérvela como evidencia.

17. Ahora, oprima el botón de ***New Query*** y transcriba el siguiente script, en el que se provoca intencionalmente un error y se invoca al bloque ***CATCH***.

```
BEGIN TRY
   DECLARE @divisor int, @dividendo int, @resultado int
   SET @dividendo = 100
   SET @divisor = 0
   SET @resultado = @dividendo/@divisor
   PRINT 'No hay error'
END TRY
BEGIN CATCH
   PRINT CONCAT('Número de error: ', ERROR_NUMBER())
   PRINT CONCAT('Error: ',ERROR_MESSAGE())
   PRINT CONCAT('Gravedad del error: ', ERROR_SEVERITY())
   PRINT CONCAT('Estado del error: ', ERROR_STATE())
   PRINT CONCAT('Línea donde ocurrió: ',ERROR_LINE())
END CATCH
```

18. Oprima el botón de ***Execute***.
19. Obtenga una imagen completa de la pantalla, mostrando el script y el resultado de la ejecución y consérvela como evidencia.
20. Ahora, oprima el botón de ***New Query*** y transcriba el siguiente script, en el que se muestra un ejemplo de manejo de excepciones con ***RAISERROR***:

```
DECLARE @tipo int, @clase int
SET @tipo = 1
SET @clase = 3
IF (@tipo = 1 AND @clase = 3)
    RAISERROR ('AVISO: El tipo no puede valer 1 si la clase es
3', 16, 1)
```

21. Oprima el botón de ***Execute***.
22. Genere una imagen que evidencie el script con el resultado de la ejecución, interprete el significado del mensaje de error.
23. Ahora, oprima el botón de ***New Query*** y transcriba el siguiente ejemplo del empleo de la cláusula **OUTPUT** para recuperar los valores que se han insertado en una tabla.

```
DECLARE @Filas TABLE
(Clave INT, Nombre VARCHAR(30), Precio MONEY, Cantidad INT)
INSERT INTO Producto(Nombre, Precio, Cantidad)
OUTPUT INSERTED.* INTO @FILAS
VALUES ('Gansito', 28, 4)
SELECT * FROM @FILAS
```

24. Oprima el botón de ***Execute***.

25. Obtenga una imagen completa de la pantalla, mostrando el script y el resultado de la ejecución y consérvela como evidencia.
26. Ahora, oprima el botón de ***New Query***, transcriba el siguiente ejemplo del uso de la cláusula ***OUTPUT*** con ***DELETE FROM***, para eliminar al mismo registro insertado previamente:

```
DECLARE @FILAS TABLE
(Clave INT, Nombre VARCHAR(30), Precio MONEY, Cantidad INT)
DELETE FROM Producto
OUTPUT DELETED.* INTO @FILAS
WHERE Nombre = 'Gansito'
SELECT * FROM @FILAS
```

27. Oprima el botón de ***Execute***.
28. Obtenga una imagen completa de la pantalla, mostrando el script y el resultado de la ejecución y consérvela como evidencia.
29. Siguiendo los ejemplos anteriores, utilice la cláusula ***OUTPUT*** para mostrar el precio anterior (DELETED.Precio) y el nuevo precio (INSERTED.Precio), para cuando se actualiza el precio del ***Bubulubu*** a ***$17.50***.
30. Obtenga una imagen completa de la pantalla, mostrando el script y el resultado de la ejecución y consérvela como evidencia.
31. Cierre la aplicación SSMS.
32. Fin de la Práctica.

Evidencias

El alumno deberá enviar al instructor las evidencias requeridas durante la realización de la práctica.

Sugerencias didácticas

El instructor deberá orientar a los alumnos que tengan dificultades en la interpretación y la realización de las instrucciones de la práctica, pero sin darles las respuestas.

Resultados

Se demuestra haber adquirido suficiente conocimiento del lenguaje ***Transact-SQL*** de SQL Server.

Bibliografía

- https://learn.microsoft.com/en-us/sql/ssms/user-assistance-in-sql-server-management-studio?view=sql-server-ver16