**FORMAS DE OBTENER DATOS DE UN XML**

La forma de extraer información de un campo XML debe ser mediante consultas XQuery, el cual ofrece muchas alternativas para obtener datos de un campo XML, entre los más comunes encontramos los siguientes:

* Estructura XML:

<parameter>

<row>1</row>

<row>2</row>

<row>3</row>

</parameter>

Ejemplo de consulta:

SELECT T.c.value('.', 'BIGINT')

FROM @xml.nodes('/parameter/row') T(c);

* Estructura XML:

<parameter>

<row Nombre="Andres" Apellidos="Martínez Rodríguez" Identificacion="159784632" />

<row Nombre="Felipe" Apellidos="Córdoba Mosquera" Identificacion="9587453" />

<row Nombre="Amy Rhousse" Apellidos="Uriana García" Identificacion="846321478" />

</parameter>

Ejemplo de consulta:

SELECT T.c.value('@Nombre', 'VARCHAR(150)'),

T.c.value('@Apellidos', 'VARCHAR(150)'),

T.c.value('@Identificacion', 'BIGINT')

FROM @xml.nodes('/parameter/row') T(c);

* Estructura XML:

<parameter>

<row>

<Nombre>Felipe</Nombre>

<Apellidos>Córdoba Mosquera</Apellidos>

<Identificacion>9587453</Identificacion>

</row>

<row>

<Nombre>Andres</Nombre>

<Apellidos>Martínez Rodríguez</Apellidos>

<Identificacion>159784632</Identificacion>

</row>

<row>

<Nombre>Amy Rhousse</Nombre>

<Apellidos>Uriana García</Apellidos>

<Identificacion>846321478</Identificacion>

</row>

</parameter>

Ejemplo de consulta:

SELECT T.c.value('Nombre[1]', 'VARCHAR(150)'),

T.c.value('Apellidos[1]', 'VARCHAR(150)'),

T.c.value('Identificacion[1]', 'BIGINT')

FROM @xml.nodes('/parameter/row') T(c);

* Estructura XML:

<parameter>

<row Tipo="M" Id="55" Nombre="Medicamentos" />

</parameter>

Ejemplo de consulta:

SELECT @xml.value('(/parameter/row/@Tipo)[1]','VARCHAR(50)'),

@xml.value('(/parameter/row/@Id)[1]','BIGINT'),

@xml.value('(/parameter/row/@Nombre)[1]','VARCHAR(50)')

* Estructura XML:

<parameter>

<row>

<Tipo>M</Tipo>

<Id>55</Id>

<Nombre>Medicamentos</Nombre>

</row>

</parameter>

Ejemplo de consulta:

SELECT @xml.value('(/parameter/row/Tipo)[1]','VARCHAR(50)'),

@xml.value('(/parameter/row/Id)[1]','BIGINT'),

@xml.value('(/parameter/row/Nombre)[1]','VARCHAR(50)')

**CLÁUSULAS WHERE EN CONSULTAS SQL SERVER**

* No se deben utilizar funciones en las cláusulas where de las consultas Sql Server.

Ejemplo: ISNULL, CONVERT, CAST, etc.

Forma incorrecta:

SELECT Codigo

FROM dbo.Medicamentos WITH(NOLOCK)

WHERE ISNULL(Codigo,'') <> ''

Forma correcta:

SELECT Codigo

FROM dbo.Medicamentos WITH(NOLOCK)

WHERE Codigo <> ''

AND Codigo IS NOT NULL

Las dos consultas retornan la misma información, pero el impacto en el rendimiento de la consulta de la forma correcta es mucho más significativo, mejorando los tiempos de respuesta del servidor de base de datos.

* Otro punto importante es realizar las comparaciones de las cláusulas where entre campos del mismo tipo. Ejemplo: si tengo un campo DATETIME y voy a realizar una condición en con ese campos, se debe comparar con un campo DATETIME.

**EJECUCION DE CONSULTAS EN CADENAS.**

Para el tipo de consultas que se realizan mediante ejecuciones de cadenas de texto, se debe utilizar el procedimiento de almacenado sp\_executesql. El cual recibe como parametro inicial el cuerpo o cadena de la consulta mediante un parametro que debe ser de tipo: ntext/ nchar/ nvarchar.

Ejemplo:

EXEC sp\_executesql N'SELECT Codigo FROM dbo.AdmConfiguracion WITH(NOLOCK)'

O bien se puede:

DECLARE @sSql NVARCHAR(1000);

SET @sSql = 'SELECT Codigo FROM dbo.AdmConfiguracion WITH(NOLOCK)';

EXEC sp\_executesql @sSql;

El llamado de estas consultas mediande el procedimiento de almacenado sp\_executesql, permite generar planes de ejecución con más probabilidades de que SQL Server los vuelva a utilizar, lo cual lo hace más eficaz.

Importante: Los parámetros de la consulta deben ser remplazados inmediatamente en la cadena y no pasarlos para que el procedimiento sp\_executesql los sustituya. Esto con el fin de que sea mas facil realizar seguimiento y mantenimiento de este tipo de consultas. Para los campos de fecha se acepta enviarlos como parametros para que el procedimiento sp\_executesql lo sustituya o reemplazarlo en la cadena de la consulta y dentro de esta declarar una variable de tipo fecha y realizar el respectivo casteo.

Ejemplos:

Sustituyendo el parametro

DECLARE @dtFecha DATETIME;

SET @dtFecha = GETDATE();

DECLARE @ParmDefinition NVARCHAR(500) = '@dtFecha DATETIME';

DECLARE @sSql NVARCHAR(1000) = 'SELECT @dtFecha AS FECHA';

EXEC sp\_executesql @sSql,@ParmDefinition, @dtFecha = @dtFecha;

O bien se puede ser:

DECLARE @sSql NVARCHAR(1000);

SET @sSql = 'DECLARE @dtFecha DATETIME; DECLARE @sFecha NVARCHAR(20) = ''20170302 10:25'' SET @dtFecha = CONVERT(DATETIME, @sFecha, 120); SELECT @dtFecha AS FECHA';

EXEC sp\_executesql @sSql;

Nota: Este tipo de consultas en lo posible debe evitarse, y solo utilizarce si es extrictamente necesario. En caso de necesitar una de estas consultas siempre debe realizarce mediante el procedimiento sp\_executesql descrito anteriormente.

**UTILIZACIÓN DE INTRUCCIONES @@IDENTITY, SCOPE\_IDENTITY() y IDENT\_CURRENT('nombre\_tabla)**

Estas tres instrucciones permiten obtener el ultimo valor de identidad insertado en una tabla, en otras palabras devuelven el valor de la llave primaria del ultimo registro insertado en una tabla, en caso de la tabla no tener llave primaria, devulven NULL.

La diferencia esta en que **@@IDENTITY** tiene en cuenta la sesión del usuario y todos los ambitos (procedimientos de almacenado, funciones, etc…)de esa sesión. **SCOPE\_IDENTITY()** tiene en cuenta la sesión del usuario y el ambito en el cual se este utilizando, ejemplo si se utiliza dentro de un procedimiento de almacenado, solo obtendra la identidad del ultimo INSERT realizado durante la ejecución de ese procedimiento. Y por ultimo **IDENT\_CURRENT('nombre\_tabla)** que a diferencia de las dos anteriores, recibe como parametro el nombre de la tabla que se quiere obtener el valor de la identidad del ultimo registro insertado, esta instrucción busca en todas las sesiones y todos los ambitos.

Es importante tener claro las diferencias y en que momento se debe utilizar cada uno de estas tres instrucciones.