Normas de Codificación

Ms SQL Server

Normas de Convención de Nombres, de Documentación de Actualizaciones, De Codificación en Transact SQL, Generales de Seguridad

Versión: 1.1.0
Proyecto
Fecha de Publicación: **13/07/2007**
Fecha de Modificación: **03/10/2022**

INFORMACIÓN DE IDENTIFICACIÓN DEL DOCUMENTO	
Creado por:	Cintia V. Gioia
Fecha de Creación:	07/07/2004
Fecha de Actualización:	03/10/2022
Primer Fecha de Publicación:	07/07/2004
Versión:	V 1.00
Fecha de Publicación:	13/07/2004
Nombre del Documento:	Normas de Codificación en SQL Server.doc
Cantidad Total de Páginas:	9
Clasificación de Seguridad:	Público
Software:	Office 365

CONTENIDOS

Normas de Convención de Nombres

La idea es utilizar una convención de nombres que posibilite una mejor y más rápida comprensión del modelo de datos de cada sistema.

- **Deben usarse nombres simples y descriptivos**.
- **Los mismos serán expresados en singular y llevarán en mayúscula la primer letra de cada palabra que lo componga**. (Banco, Comercio, LiquidacionEstablecimiento, Usuario, etc) Esto es llamada notación Pascal.

Convención de Nombres de Base de Datos (**)

Las ***bases de datos*** se llamarán **DB_<NombreBaseDatos>**. (Solo en este caso pueden usarse plurales)

Ejemplos: DB_Monitor, DB_Produccion

Convención de Nombres de Tablas (**)

Los nombres de las ***tablas*** deben identificar claramente a la entidad lógica que representan y deben respetar la convención general y estar expresadas en singular y llevar en mayúscula la primer letra de cada palabra que lo componga.

Ejemplos: Banco, Sucursal, Comercio, Usuario, Tarjeta, Cupon, ProcesoLiquidacion

Convención de Nombres de Campos de Claves Primarias e Índices asociados (**)

La ***clave primaria*** de cada entidad será un entero con el nombre **<NombreTabla>ID**. Se recomienda la utilización de identities globales por base de datos.

El nombre del **índice** correspondiente a la clave primaria deberá llamarse **PK_<NombreTabla>.**

Ejemplo: La clave primaria de la tabla Empleado será el campo EmpleadoID y será denomidada PK_Empleado.

En el caso de querer utilizar **Global ID** ver Anexo 1 (GlobalID)

Convención de Nombres de Campos (**)

Los nombres de los *campos* en singular y llevar en mayúscula la primera letra de cada palabra que lo componga (notación pascal). Ejemplo: **RazonSocial**

Convención de Nombres de Unique Constraints (**)

Los ***unique constraints*** se denominarán con el formato:
UQ_<NombreCampo1>_<NombreCampo2>_ < . . . >

Ejemplos:
Si en la tabla Alumno el campo cDNI es único, existirá un índice denominado UQ_cDNI.

En cambio, si en la tabla RenglonesFactura la concatenación de los campos cCodigoFactura, cRenglonFactura debe ser una clave única, existirá un índice UQ_cCodigoFactura_ cRenglonFactura.

Convención de Nombres de Check Constraints (**)

Las restricciones o ***check constraints*** se denominarán con el formato:
CK_<NombreCampo1>_<NombreCampo2>_ < . . . >

Convención de Nombres de Campos Foreign Keys e Indices asociados (**)

En cuanto a las referencias a otras tablas o ***foreign keys*** el formato a utilizar será el siguiente:
Las relaciones entre tablas se harán siempre referenciando a los primary key.
El nombre del campo se denominará como el primary key al que referencia.
El índice se llamará FK_<NombreTabla1>_<NombreTabla2>

Ejemplos:
Para saber a qué Banco pertenece cada Sucursal, en esta segunda tabla se hará referencia a la de Banco; con lo cuál en Sucursal habrá un campo BancoID y una relación denominada FK_Sucursal_Banco.

Convención de Nombres de Tablas de Relación

Las ***tablas de relación*** (las que traducen las relaciones lógicas “muchos a muchos”) se llamarán **<NombreTabla1><NombreTabla2> y la clave primaria estará compuesta por la concatenación de las claves primarias de ambas tablas** (aquí no habrá un campo ID propio)

Ejemplos:
Si un Médico puede atender muchas Obras Sociales y a la vez las Obras Sociales tienen en cartilla a muchos Médicos; quedarían las tablas Médico, ObraSocial y MedicoObraSocial. Esta última tabla identificará las relaciones entre médicos y obras sociales, y contendrá entre otros, los campos MedicoID y ObraSocialID.
Existirán también los índices:
PK_ MedicoObraSocial,

FK_ MedicoObraSocial _Medico,
FK_ MedicoObraSocial _ObraSocial.

Convención de Nombres de Procedimientos (**)

Los ***procedimientos almacenados*** se denominarán **sp_<NombreStoredProcedure>**

Ejemplos: sp_ProcesamienDeFactura, sp_AltaEmpleado

Convención de Nombres de Vistas (**)

Las ***vistas*** se denominarán **v_<NombreView>**

Ejemplos: v_Sueldos, v_Gastos

Convención de Nombres de Triggers(*)

Los ***Triggers*** que se utilicen deberán respetar el siguiente patrón:

- Todos los Triggers deben comenzar con las letras TG.
- Luego se agregará una 'I' si el trigger es de insert, una 'U' si es de update y una 'D' si es de delete.
- Estas letras pueden sumarse, ya que un mismo Trigger puede utilizarse para Insert y Update por ejemplo, se utilizará entonces TGUI. Luego seguirá un '_' (underscore o guión bajo) y el nombre del trigger.

Ejemplos: TGI_Sector, TGU_Pagos, TGD_Empleados

Convención de Nombres de Campos Desnormalizados

El modelo de datos debe estar al menos en tercera forma normal. En caso de que algún dato esté desnormalizado deberá explicarse claramente el porque e indicar el nombre del campo con x_

Ejemplos: x_iImportePlan, x_cUsuarios

Normas de Documentación de Actualizaciones de Procedimientos Almacenados(*)

Todos los stored procedures deberán contener un comentario que especifique:

-- Autor:
-- Fecha de Creación:
-- Función del Stored:
-- Task que lo ejecuta (Frecuencia, hora) o Sistema que lo utiliza:
-- Servidor:
-- Base de Datos:
-- Usuario:

Normas Generales de codificación en Transact SQL

Escritura de Sentencias SQL(*)

Las sentencias SQL deberán escribirse de la siguiente manera:

- Los comandos y modificadores propios de la sintaxis SQL deberán escribirse en mayúscula.
- Los nombres o alias de campos, tablas u otros objetos tal como se crearon.

Ejemplos:

SELECT cDni FROM Empleado WHERE dFechaNacimiento > '19760704'

sp_CalcularDescuento(iPuntos)

Acceso a Datos

El acceso a los datos deberá siempre realizarse a través de vistas o stored procedures.

De esta manera se logra que un cambio en el modelo no afecte las aplicaciones de una forma significativa, ya que cambiando la vista o stored procedure no hace falta cambiar las aplicaciones. De otra manera un cambio en el modelo provoca numerosos cambios en las aplicaciones. Todo esto sin mencionar que de esta forma es más sencillo manejar eficientemente los diferentes permisos y el control del acceso a los datos. Obviamente que es importante lograr mantener la interfaz o firma de la vista o stored procedure.

Normas Generales de Seguridad

En cuanto a los Usuarios de las bases y los permisos:

Manejarse por grupo (perfiles) y no por usuario, así facilitamos la administración de los mismos y logramos un mejor control en cuanto al acceso a los datos.

Los usuarios de las aplicaciones deben ser creados específicamente para las aplicaciones para lograr así que solo se le den los permisos necesarios.

La seguridad deberá administrarse en forma integrada con Windows NT, poniendo a cada usuario en los grupos que corresponda de acuerdo a sus necesidades. De esta manera se simplifica la administración de usuarios, así como la auditoría de las operaciones.

Anexo 1 (GlobalID)

Para la implementación de GlobalID sugerimos utilizar la siguiente tabla y los Stored Procedures que siguen.

Tabla GlobalID:

```
CREATE TABLE GlobalID (
        iNextID INT NOT NULL
)
GO
```

Stored Procedure que devuelve el siguiente GlobalID a asignar:

```
CREATE PROCEDURE sp_GetNewID AS
SET NOCOUNT ON

BEGIN TRANSACTION

                DECLARE @iNextID  INT

SELECT @iNextID = iNextID FROM GlobalID

        UPDATE GlobalID SET iNextID = @iNextID + 1

SELECT @iNextID

COMMIT TRANSACTION

GO
```

Stored Procedure que genera un Query que tiene por objetivo actualizar el GlobalID al máximo valor que exista en la DB:

```
CREATE PROCEDURE sp_SetMaxGlobalID AS
SET NOCOUNT ON

DECLARE @cTableName SYSNAME
DECLARE @cCommand   VARCHAR(254)
DECLARE TableCursor CURSOR FOR
SELECT
       SysObjects.name
FROM
       SysObjects INNER JOIN
       SysColumns ON SysObjects.id = SysColumns.id
WHERE
       SysObjects.type = 'U'           AND
       SysColumns.name = SysObjects.name + 'ID'
ORDER BY
       SysObjects.name

PRINT '--Calculating GlobalID'
PRINT 'SET NOCOUNT ON'
PRINT 'DECLARE @iTempID INT'
PRINT 'DECLARE @iMaxID  INT'

PRINT 'SELECT @iTempID = 0'
PRINT 'SELECT @iMaxID  = 0'

OPEN TableCursor

FETCH NEXT FROM TableCursor INTO
       @cTableName

WHILE (@@FETCH_STATUS <> -1)
BEGIN
       IF (@@FETCH_STATUS <> -2)
       BEGIN
              SELECT @cCommand = 'SELECT @iTempID = MAX(' + @cTableName + 'ID) FROM ' +
@cTableName
              PRINT @cCommand
              PRINT 'IF (@iMaxID < @iTempID)'
              PRINT 'BEGIN'
              PRINT ' SELECT @iMaxID = @iTempID'
              PRINT 'END'


              FETCH NEXT FROM TableCursor INTO
                     @cTableName
       END
END
CLOSE TableCursor
PRINT 'SELECT @iMaxID iMaxGlobalID'
GO
```