**Estándares para SQL SERVER**

**Historia del Documento**

|  |  |  |  |
| --- | --- | --- | --- |
| **Versión** | **Descripción** | **Autor** | **Fecha** |
| 1.0 | Versión original | Juan Portero López | 10/04/2017 |
| 1.1 | Aumento de la columna **eliminado** | Juan Portero López | 21/04/2017 |
|  |  |  |  |
|  |  |  |  |

**Contenido**

[1. Versión de SQL Server 2](#_Toc480532237)

[2. Nomenclatura de nombres 2](#_Toc480532238)

[3. Campos de auditoria y borrado lógico 3](#_Toc480532239)

[4. Campos secuenciales 3](#_Toc480532240)

[5. Tipos de Campos 4](#_Toc480532241)

[6. Longitud de Campos y uso 4](#_Toc480532242)

[1. Cadena de caracteres 5](#_Toc480532243)

[2. Valores numéricos 5](#_Toc480532244)

[3. Tipo documentos e imágenes 5](#_Toc480532245)

[7. Nombre de base de datos y agrupación 5](#_Toc480532246)

[8. Mejores practicas 5](#_Toc480532247)

[9. Misceláneos 7](#_Toc480532248)

A continuación, se detallan algunas reglas generales para la nomenclatura en los objetos de la base de datos.

# Versión de SQL Server

|  |  |
| --- | --- |
| Versión | SQL Server 2014 enterprise de 64 bits |
| Idioma | Español España Unicode |
| Intercalación (Collate) | Modern-Spanish, case-insensitive, accent-sensitive, kanatype-insensitive, width-insensitive |

# Nomenclatura de nombres

1. Usar “**minusculas**” para nombrar objetos de SQL Server como tablas, vistas, procedimientos almacenados, funciones, columnas, esquemas.
2. Si son nombre compuesto debe ser separados por el carácter ‘\_’
3. Tablas y vistas deben ser nombradas en singular sin finalizar con "s".
4. Los nombres de las tablas deben reflejar de manera precisa su contenido y función.
5. Debe existir un PRIMARY KEY en todas las tablas de la base de datos con una columna de nombre id\_<nombre\_de\_la\_tabla>. Es una práctica común utilizar la clave primaria como columna IDENTITY.

Ejemplo: id\_persona

1. No se deben utilizar espacios en el nombre.
2. Los nombres de las vistas siguen las mismas convenciones de los nombres de las tablas, con prefijo vis\_.
3. No se deben utilizar palabras reservadas como nombres de tablas y campos.
4. Para el nombre de procedimientos, funciones y vistas deben tener el esquema un prefijo más el nombre en formato entidad\_transaccion:

|  |  |  |
| --- | --- | --- |
| **Objeto** | **Prefijo** | **Ejemplo** |
| Procedimientos almacenados | proc\_ | seguridad.proc\_usuario\_insertar |
| Funciones | func\_ | seguridad.func\_usuario\_validar |
| Vistas | vis\_ | seguridar.vis\_usuario\_sesiones |

1. Prefijos para otros objetos de base de datos. Usar el siguiente patrón para nombres de:
   1. Triggers: TR\_<Nombre de la tabla>\_<acción><descripción>

Ejemplo:

TR\_usuario\_update\_nombre\_usuario

* 1. Indices : IX\_<Nombre de la tabla>\_<columnas separadas por\_>

Ejemplo:

IX\_usuario\_nombre

* 1. Primary Key : PK\_<Nombre de la tabla>

Ejemplo:

PK\_usuario

PK\_oficina

* 1. Foreign Key : FK\_<Nombre tabla\_1>\_<Nombre tabla\_2>

Ejemplo:

FK\_oficina\_usuario

* 1. Default: DF\_<Nombre de la tabla>\_<Nombre columna>

Ejemplo:

DF\_usuario\_nombre

# Campos de auditoria y borrado lógico

Para identificar quien creo y realizo el último cambio en el registro se tiene los siguientes campos:

|  |  |  |
| --- | --- | --- |
| **Nombre Campo** | **tipo** | **Uso de campo** |
| eliminado | bit | Usado para borrado lógico.  -1 el registro está borrado.  -0 el registro no está borrado |
| creacion\_usuario | varchar(20) | Usuario que crea el registro. |
| creacion\_fecha | datetime | La fecha de creación del registro |
| creacion\_equipo | varchar(20) | Dirección física(mac address) del equipo desde donde se creó el registro |
| modifica\_usuario | varchar(20) | Usuario que modifica el registro |
| modifica\_fecha | datetime | Fecha de modificación del registro |
| modifica\_equipo | varchar(20) | Dirección física(mac address) del equipo desde donde se modificó el registro |

Todos los campos son obligatorios. EL campo creacion\_fecha tiene el valor por omisión con la función getdate(). El resto de campos deben ser llenados durante el ingreso de los registros, en modifica\_usuario con el mismo valor de creacion\_usuario y modifica\_fecha con el mismo valor de creacion\_fecha.

# Campos secuenciales

En la tablas **[comun].[secuencial]** se registra el nombre del campo y el ultimo valor asignado.

El procedimiento almacenado **[comun].[proc\_siguiente\_secuencial]** tiene la lógica para generar el nuevo secuencial o inicializar el primer valor.

# Tipos de Campos

|  |  |  |
| --- | --- | --- |
| **Tipo** | **Uso** | **Detalle** |
| varchar | Cadena de caracteres | Varchar(200) para campos que almacenen descripciones y valor por omisión.  Varchar(50) Para nombres de conceptos por ejemplo:   * Décimo tercer sueldo * Intereses de mora   Varchar(20) para campos que almacenan código, por ejemplo   * Código de cuentas contables * Codigos de préstamos   Varchar(max) para campos que almacenen información XML.  Se puede almacenar cualquier texto en el campo, incluso si no es válido xml documentos. Conflictos con otro set de caracteres genera conflictos. |
| int | Claves primarias |  |
| money | Campos que almacenen valores monetarios |  |
| datetime | Campos de auditoria y otros que se requieran hora |  |
| bit | Para condiciones de verdadero o falso |  |
| [varchar(max)](https://msdn.microsoft.com/es-es/library/ms176089.aspx) y [varbinary(max)](https://msdn.microsoft.com/es-es/library/ms188362.aspx) | Para texto e imágenes | Recomendado por Microsoft para nuevos desarrollos. |

# Longitud de Campos y uso

* Todos los campos no deben permitir valores nulos. Si algún campo requiere valores nulos se lo debe justificar documentadamente.
* En los campos no se permiten valores por omisión. Si algún campo requiere valores por omisión se lo debe justificar documentadamente.

## Cadena de caracteres

* Los campos tipos **varchar** tendrán un valor por omisión de 200 caracteres
* Nombres de conceptos son de tipo **varchar(50)**
* Nombres de campos de descripciones se mantienen con el valor omisión o modificado según las necesidades
* Todos los campos deben tipo fecha deben ser definidos como **datetime**.
* Para códigos de cuentas contables será una campo **varchar(20)**

## Valores numéricos

|  |  |  |
| --- | --- | --- |
| **Uso** | **Tipo** | **Comentario** |
| Valores números | Decimal (18,4) | Facilidad para cálculos  Se usa para porcentajes, tasas. Valores deben ir entre 0 y 100.  Para 15,5% se debe grabar 15.5 y no 0.155 |
| Valores monetarios | Money |  |

## Tipo documentos e imágenes

* Todos los documentos e imágenes se debe usar tipo de dato [**varchar(max)**](https://msdn.microsoft.com/es-es/library/ms176089.aspx) y [**varbinary(max)**](https://msdn.microsoft.com/es-es/library/ms188362.aspx).
* Para archivos XML se usa tipo de datos **varchar(MAX).**

# Nombre de base de datos y agrupación

1. La base de datos se llamará **siisspolweb**
2. Los objetos de la base de datos serán agrupados usando **esquemas**.

Ejemplo para el esquema “seguridad”:

seguridad.oficina

seguridad.oficina\_usuario

# Mejores practicas

1. En lo posible, normalizar la estructura de la base de datos basada en el 3er forma de normalización. La normalización es el proceso de diseñar un modelo de datos para almacenar eficientemente los datos en una base de datos.
2. Evitar el uso de **SELECT** \* en consultas SQL. Escriba todas las columnas requeridas en la consulta después del SELECT.

Ejemplo:

SELECT oficina,nombre,estado

FROM seguridad.usuario

1. Evitar el uso de tablas temporales y tablas derivadas estos consumen más lecturas a disco. Como buena práctica usar **CTE** (Common Table Expression).

Por ejemplo:

-- Definir cte

WITH cte\_usuario (id\_usuario, nombre)

AS

(

SELECT id\_usuario, nombre

FROM seguridad.usuario

)

-- Consulta sobre cte.

SELECT nombre

FROM cte\_usuario

1. Usar **SET NOCOUNT ON** al inicio de SQL scripts (batches), Stored Procedures y Triggers. Esto mejora el rendimiento de Stored Procedure.
2. Usar un **formato** apropiado en consultas SQL usando indentación.

Ejemplo: Mal formato

SELECT direccion, nombre FROM oficina of INNER JOIN usuario us ON us.id\_oficina = of.id\_oficinaID

Ejemplo: Correcto formato

SELECT direccion, nombre

FROM oficina of

INNER JOIN usuario us ON us.id\_oficina = of.id\_oficina

1. Usar mayúsculas para todas las **palabras clave** de SQL.
2. Ejemplo:
3. SELECT, UPDATE, INSERT, WHERE, INNER JOIN, AND, OR, LIKE.
4. Columnas con **Default value** constraint no debe permitir valores NULLs.
5. Usar la **PRIMARY** key en la condición **WHERE** de la sentencia UPDATE o DELETE esto deberá evitar posibles errores y mejora el rendimiento.
6. Siempre crear procedimientos almacenados en la misma base de datos donde su tabla pertinente existe de lo contrario se reducirá el rendimiento de la red.
7. Evitar como sea posible el uso de cursores en el lado del servidor. Use variables tipo tablas o CTE y navegue a través de un WHILE por los registros.

Ejemplo:

DECLARE @total INT,@contador INT, @idusuario VARCHAR(20),@nombre VARCHAR(50)

DECLARE @tblusuario TABLE

(id [INT] IDENTITY(1,1) NOT NULL,

[id\_usuario] [varchar](20) NOT NULL,

[nombre] [varchar](50) NULL)

INSERT INTO @tblusuario(id\_usuario,nombre)

SELECT id\_usuario,nombre

FROM [seguridad].[usuario]

SELECT @total=MAX(id)

FROM @tblusuario

SET @Contador=1

WHILE @Contador <=@total

BEGIN

SELECT @idusuario=id\_usuario,

@nombre=nombre

FROM @tblusuario

WHERE id =@contador

SELECT @nombre

SELECT @Contador=@Contador+1

END

1. En las sentencias **if exists** no usar select \* from es mejor usar select (1)
2. En las sentencias **inserts** siempre se debe especificar los nombres de los campos

# Misceláneos

1. Para programación usar el tipo de caracteres UTF-8
2. Manejar una tabla de sesiones para usuarios
3. Para propósitos de auditoria usar trazas calendarizadas diariamente con los apropiados filtros.
4. Comentarios en procedimiento almacenados y funciones. Para llevar control de los cambios se debe agregar una cabecera con comentarios e información histórica de los cambios. Por ejemplo:

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

/\*

Nombre: proc\_usuario\_insertar

Descripción: crea un registro en la tabla usuario

\*\*\*\* Histórico \*\*\*\*

Fecha siglas/nombre Descripción

13/04/2017 Juan Portero Primera versión

\*/

CREATE PROCEDURE [seguridad].[proc\_usuario\_insertar]

@AS\_USU\_USUARIO VARCHAR(20),

@AI\_ID\_OFICINA INT,

@AI\_ID\_PERSONA INT,

@AS\_ESTADO int,

@AS\_EMAIL VARCHAR(100) ,

…………………….

1. Configurar Microsoft JDBC Driver para soportar plataforma Java. Seguir el (XA transaccional) procedimiento según el enlace: <https://docs.microsoft.com/en-us/sql/connect/jdbc/understanding-xa-transactions>

|  |  |  |
| --- | --- | --- |
| Elaborado por: | Juan Portero López |  |
| Revisado por: | Carlos Córdova |  |
| Aprobado por: |  |  |